Tufts UNIVERSITY

COMP 150-04

User Interfaces for Mobile Platforms

Instructor:

Karen Donoghue, MS

Teaching Assistant:

Aaron Wishnick

Instructor: Karen Donoghue,
karen@humanlogic.com
SKYPE: KDONOGHUE (up until 9pm EST)

TA: Aaron Wishnick,
Aaron_B.Wishnick@tufts.edu

Class Dates/Time/Location:
Thursdays 6:30p-9:00p
Room: Tisch 313 (Library)

Office Hours
by Appointment

Applications

Calendar

Phone

Weather

Maps

UI components

Button

List

Combo

Theme

OS Services

Location Data

Streaming Video

Core OS

Bluetooth

USB

Hardware

Camera

Sensors

Display

Keypad

Class #2 Agenda

- Review Android app navigation structures
- Assignment 1 discussion
- Review of Android UI Design/Components
- Discuss Assignment #2
- Setting up Android Studio

Apps deconstructed (Android)

Navigation structure

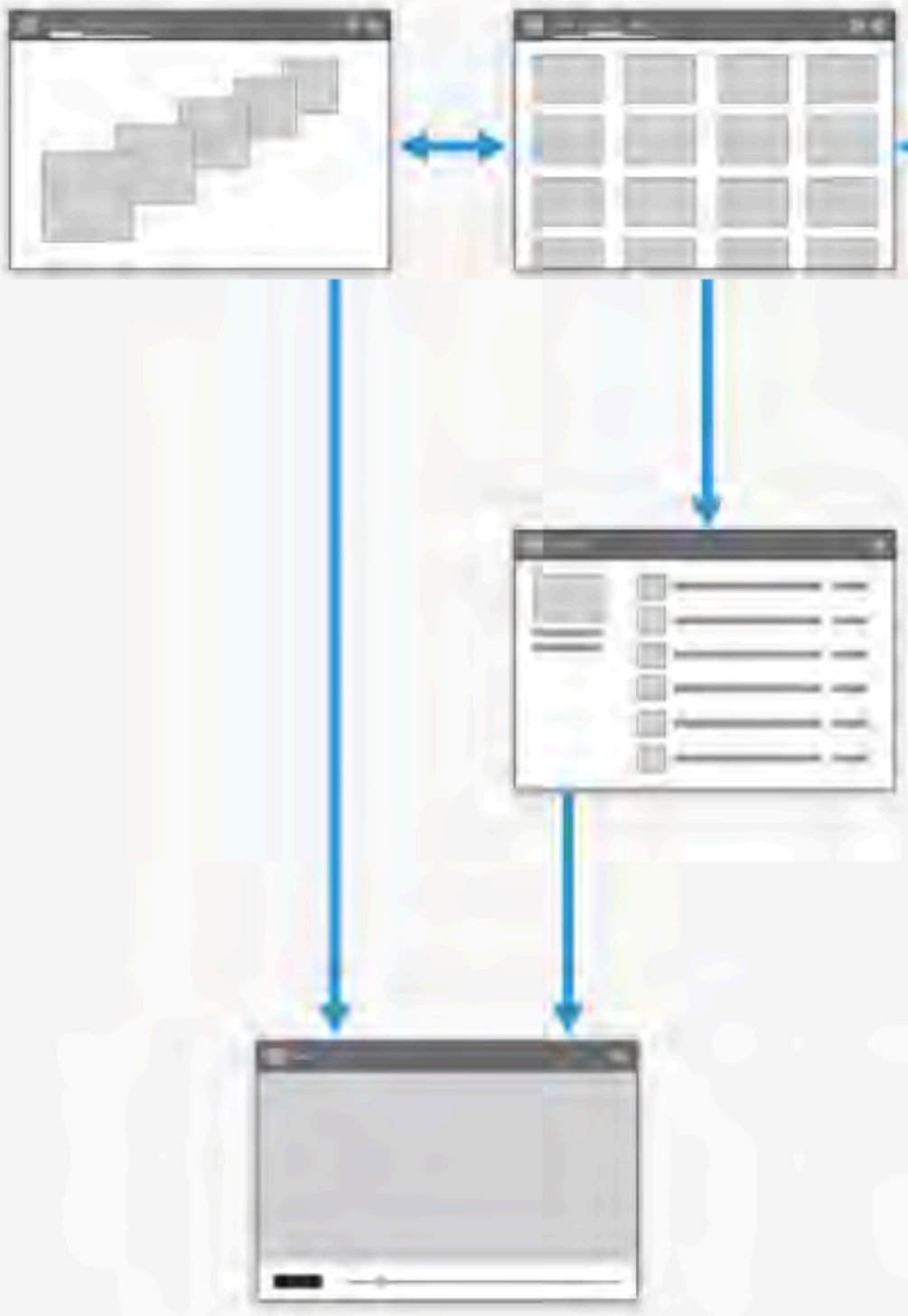
- Please view this video over the next week:
- **Structure in Android App Design**
- https://developers.google.com/events/io/sessions/326301704
- PDF for reference:
  - 118 - I_O 2013- Structure In Android App Design.pdf

Portions of these pages are reproduced from work created and shared by the Android Open Source Project and used according to terms described in the Creative Commons 2.5 Attribution License.

General Structure

See http://developer.android.com/design/patterns/app-structure.html

A typical Android app consists of top level and detail/edit views. If the navigation hierarchy is deep and complex, category views connect top level and detail views.



Top level views

The top level of the app typically consists of the different views that your app supports. The views either show different representations of the same data or expose an altogether different functional facet of your app.

Category views

Category views allow you to drill deeper into your data.

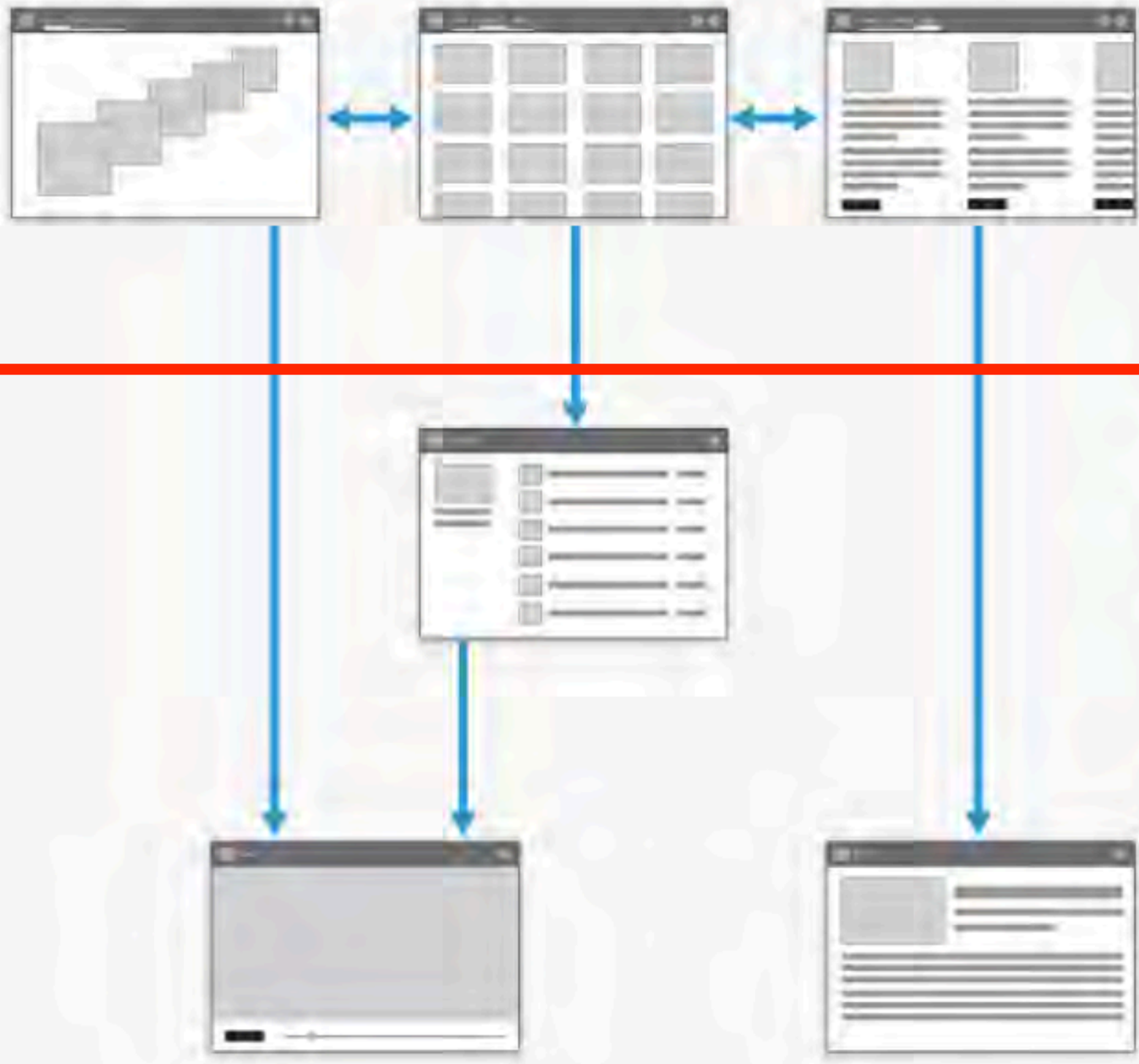
Detail/edit view

The detail/edit view is where you consume or create data.

General Structure

See http://developer.android.com/design/patterns/app-structure.html

A typical Android app consists of top level and detail/edit views. If the navigation hierarchy is deep and complex, category views connect top level and detail views.



Top level views

The top level of the app typically consists of the different views that your app supports. The views either show different representations of the same data or expose an altogether different functional facet of your app.

Category views

Category views allow you to drill deeper into your data.

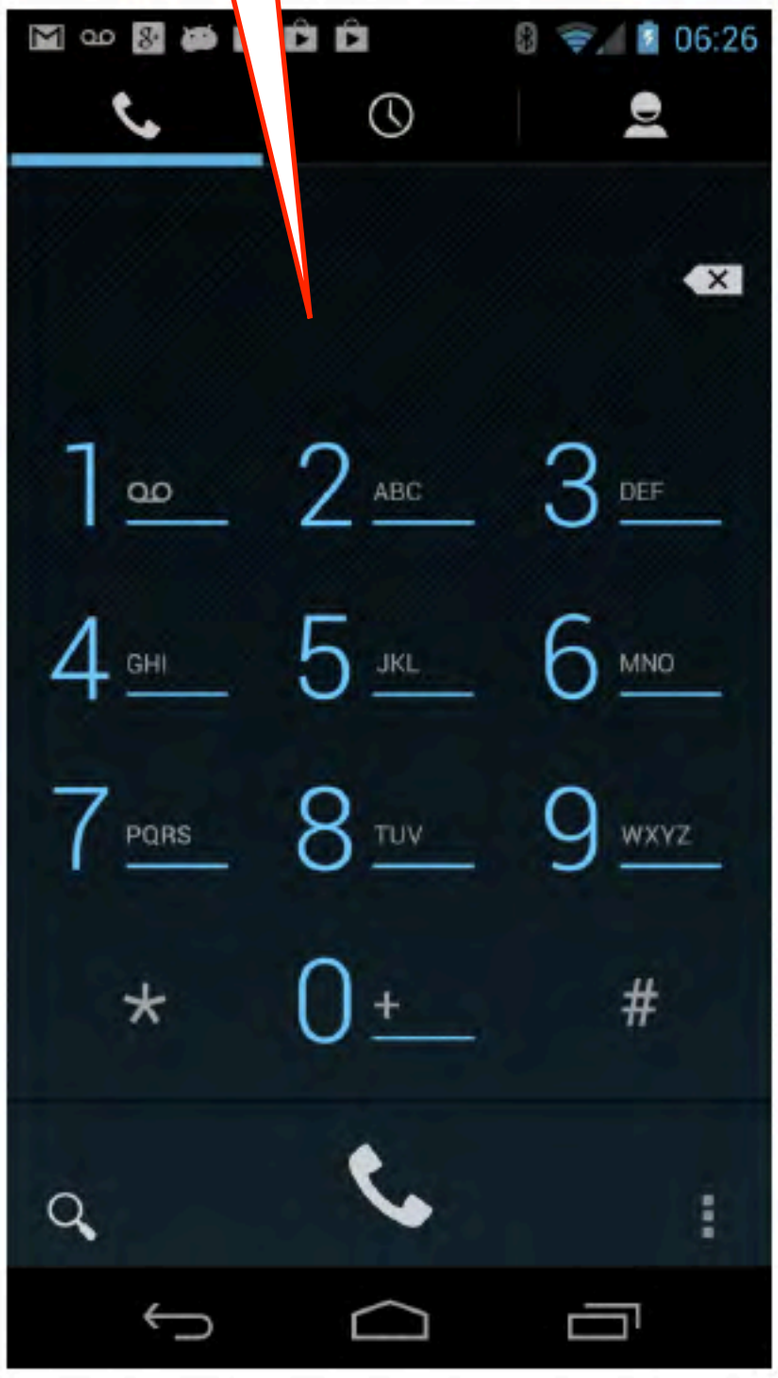
Detail/edit view

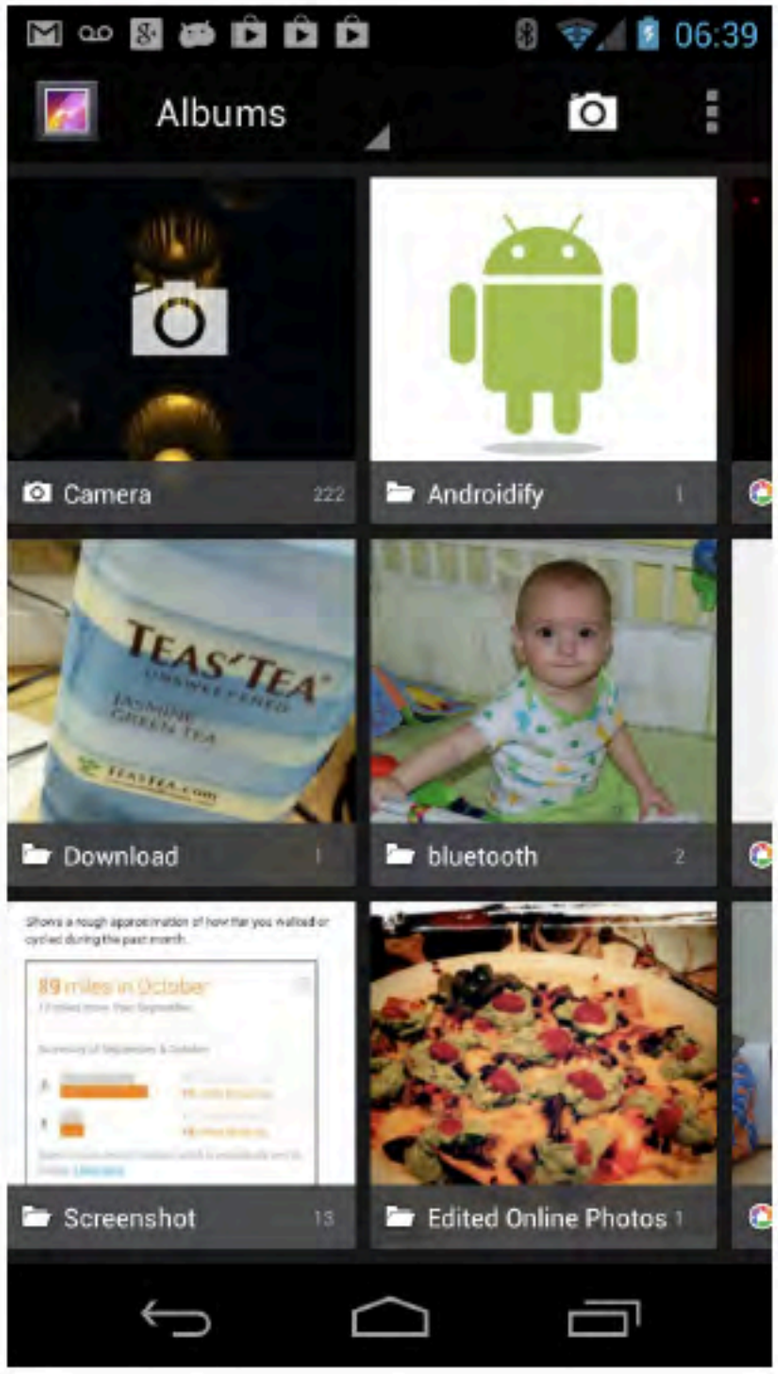
The detail/edit view is where you consume or create data.

Communicates the app's primary purpose

"I am a calling app"

Top level





General Structure

See http://developer.android.com/design/patterns/app-structure.html

A typical Android app consists of top level and detail/edit views. If the navigation hierarchy is deep and complex, category views connect top level and detail views.

Top level views

The top level of the app typically consists of the different views that your app supports. The views either show different representations of the same data or expose an altogether different functional facet of your app.

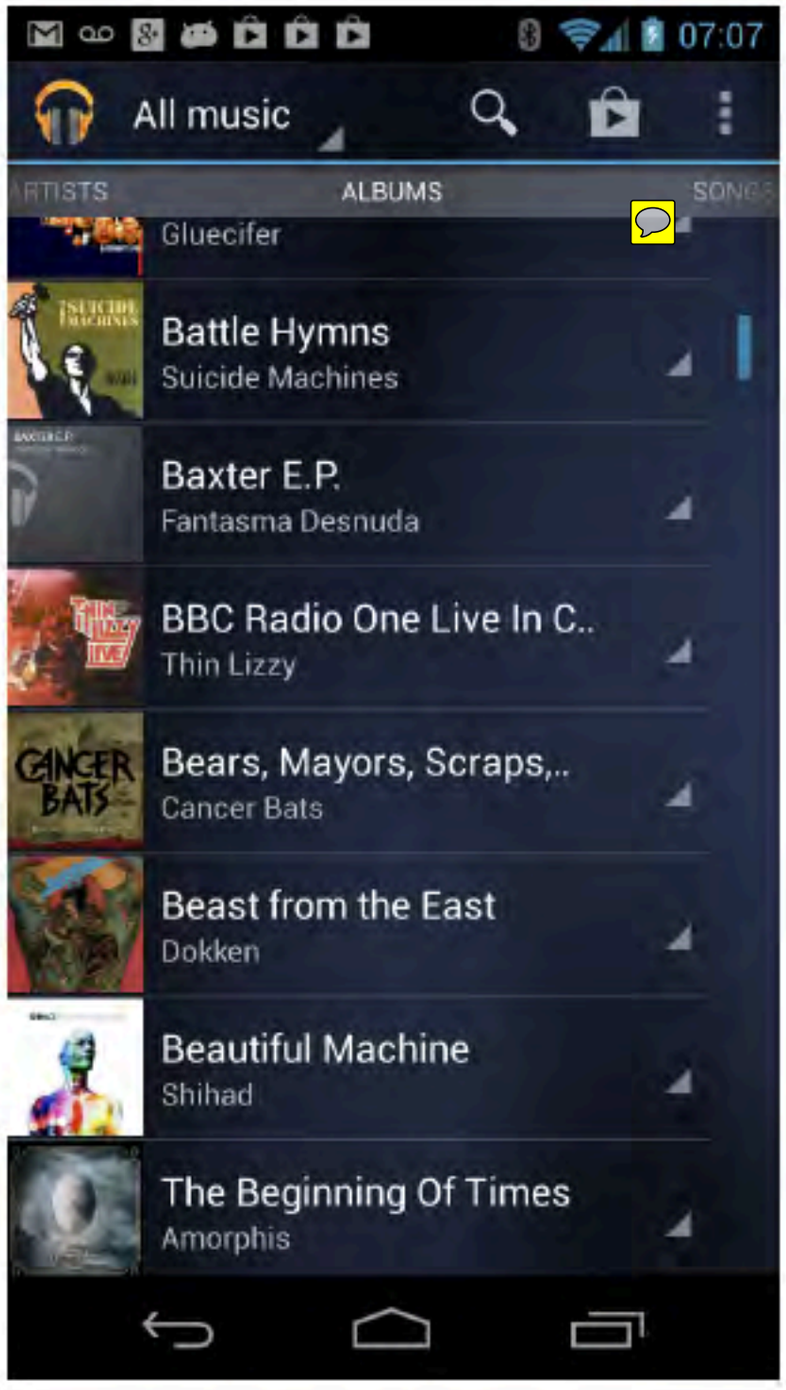
Category views

Category views allow you to drill deeper into your data.

Detail/edit view

The detail/edit view is where you consume or create data.

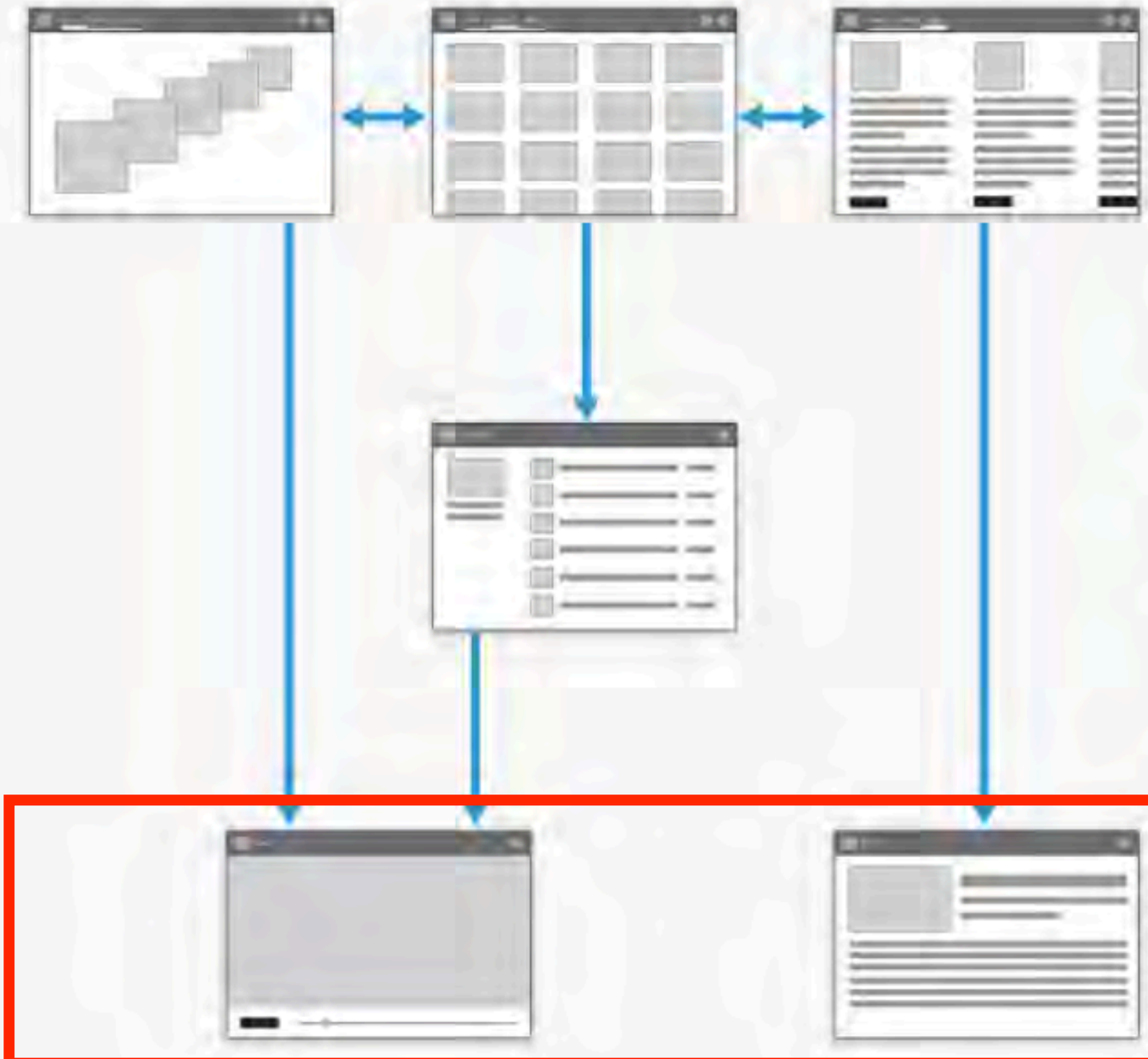
Category Views



General Structure

See http://developer.android.com/design/patterns/app-structure.html

A typical Android app consists of top level and detail/edit views. If the navigation hierarchy is deep and complex, category views connect top level and detail views.



Top level views

The top level of the app typically consists of the different views that your app supports. The views either show different representations of the same data or expose an altogether different functional facet of your app.

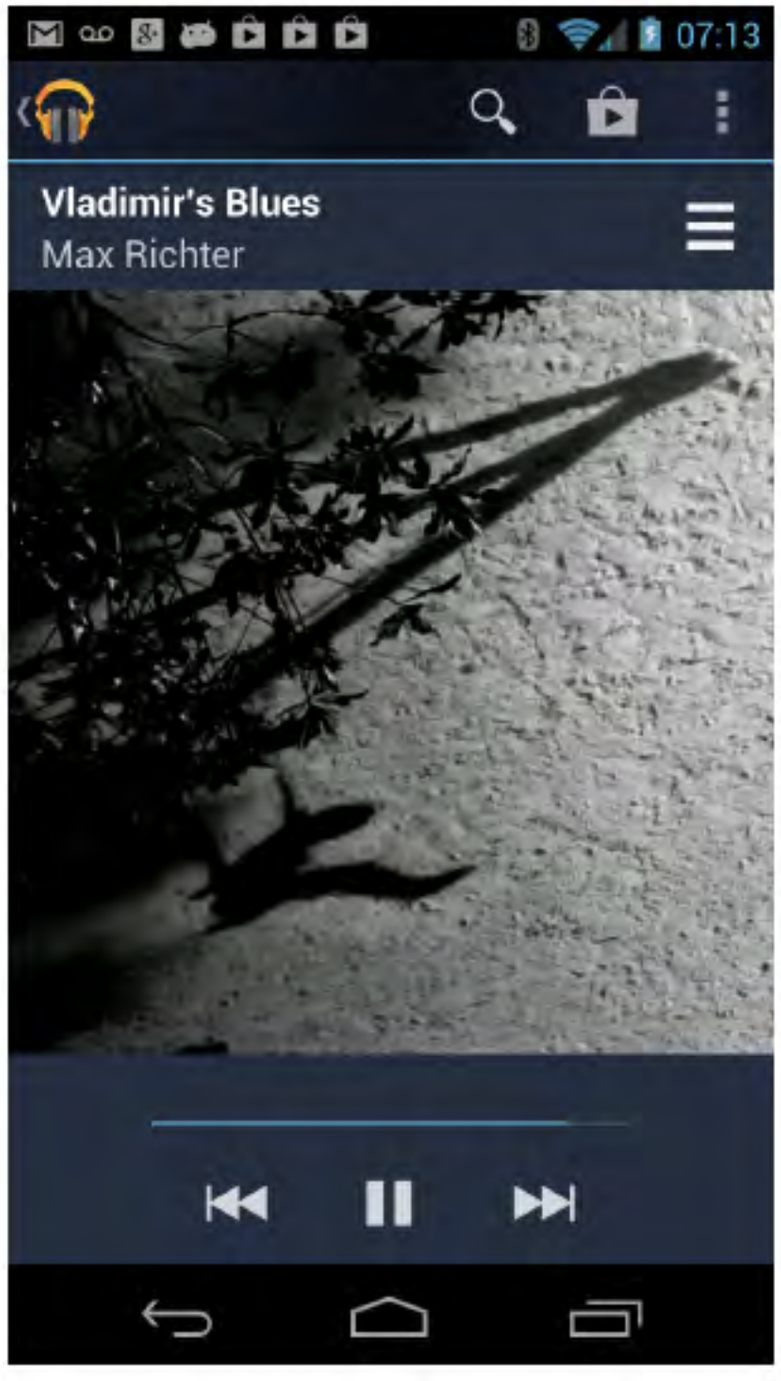
Category views

Category views allow you to drill deeper into your data.

Detail/edit view

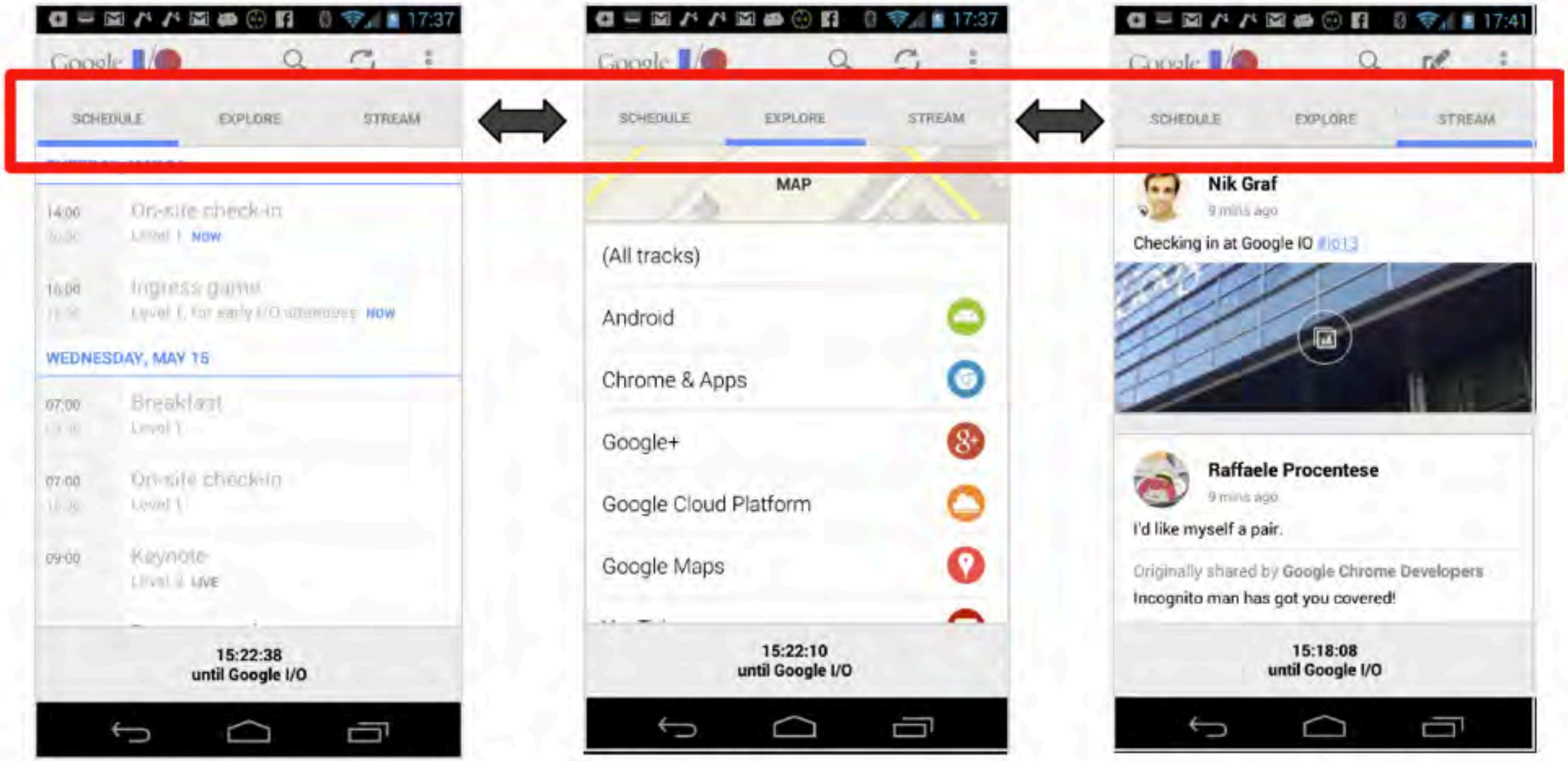
The detail/edit view is where you consume or create data.

Detail/Edit view



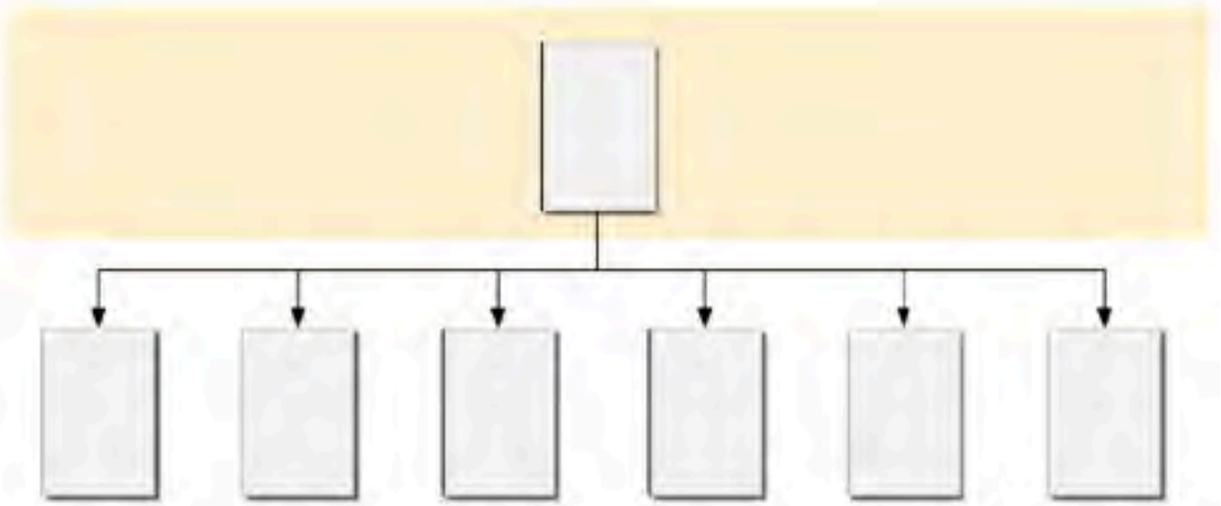
Design Patterns for Navigation controls

Top-level navigation

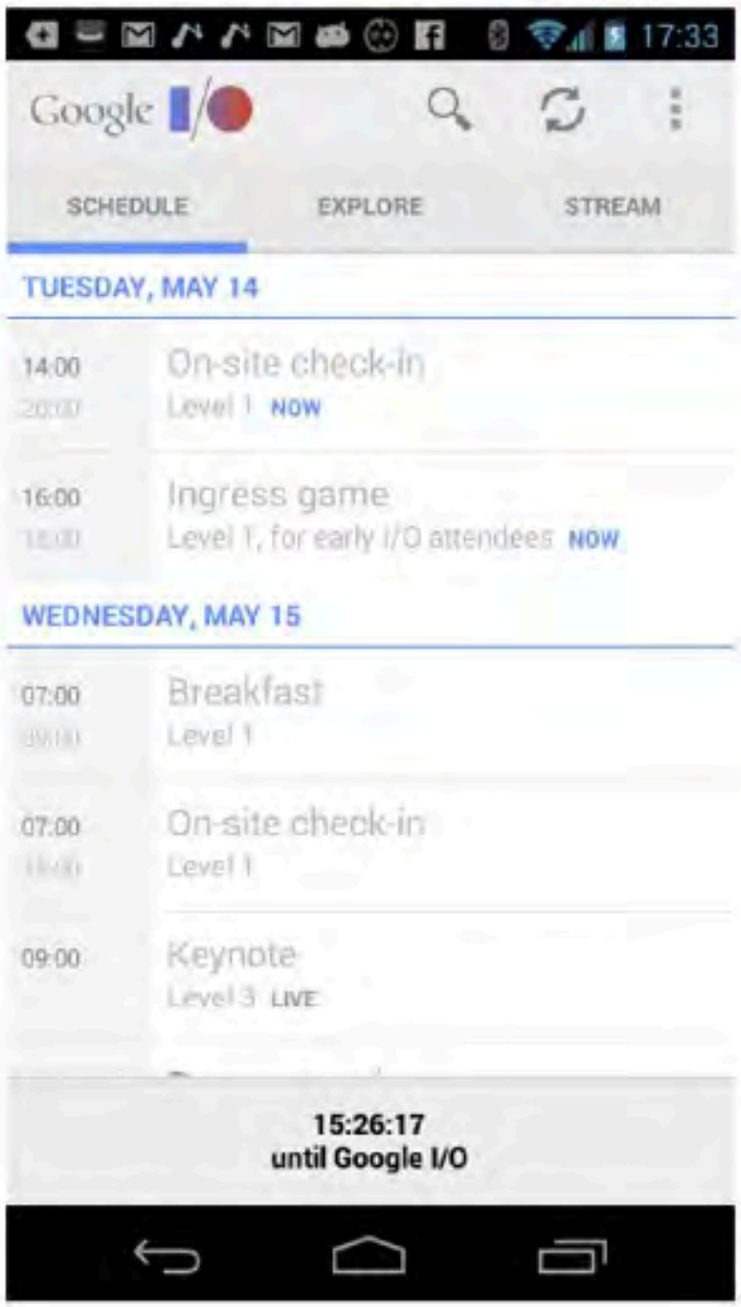


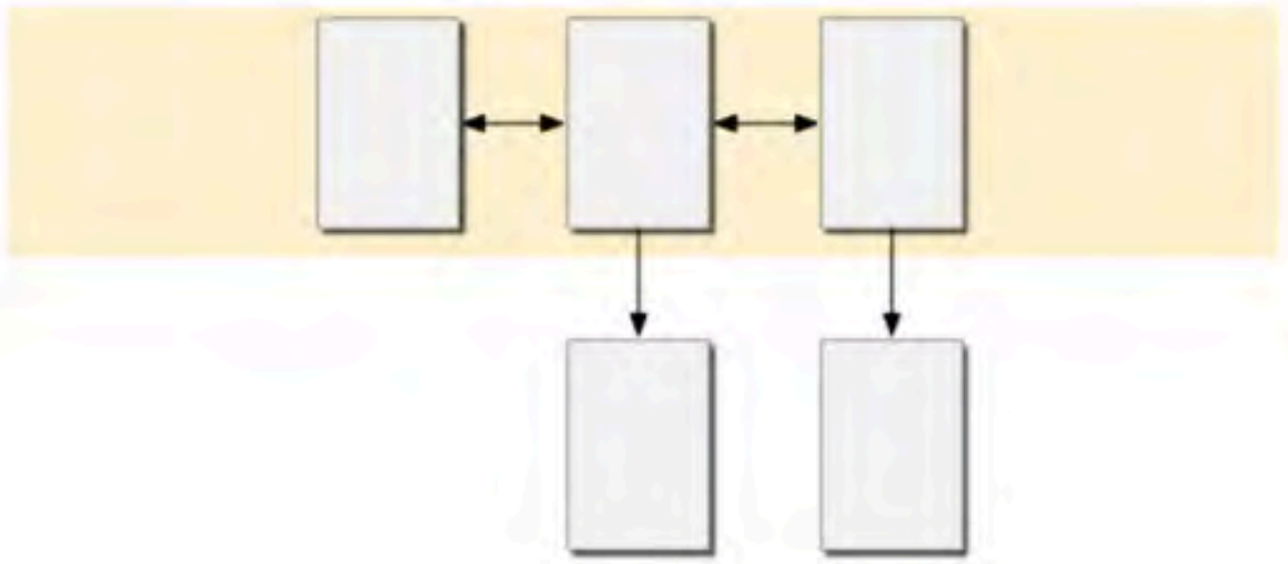
Six Pack

2:30
Travel Mania
Hotels
Flights
Cars
Cruises
Last minute
My account

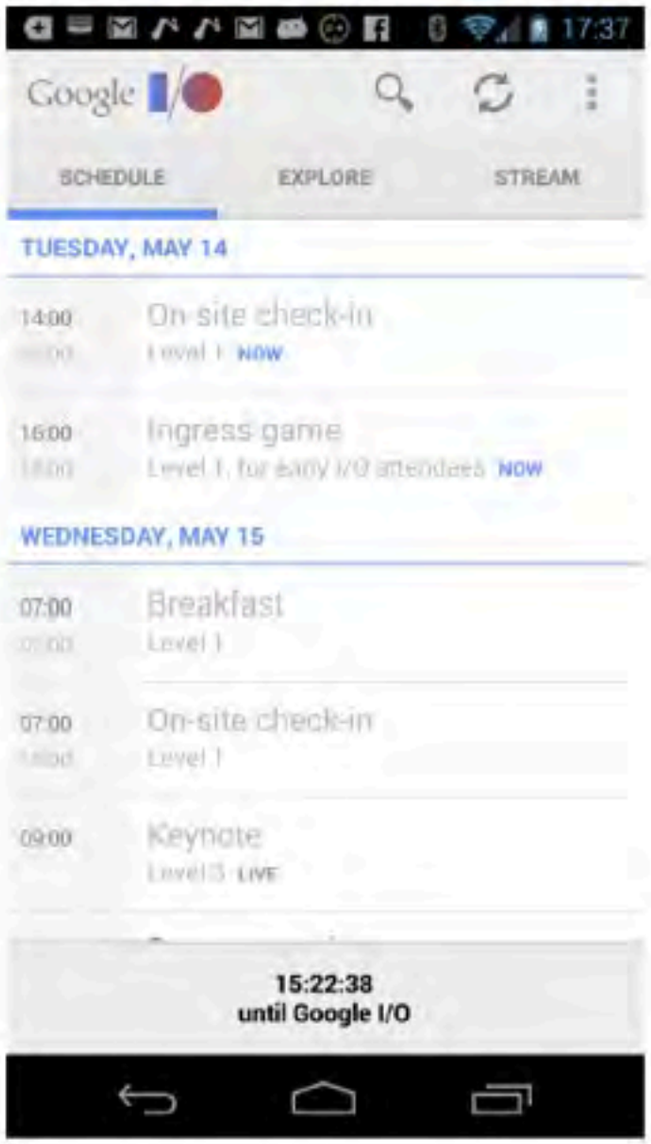


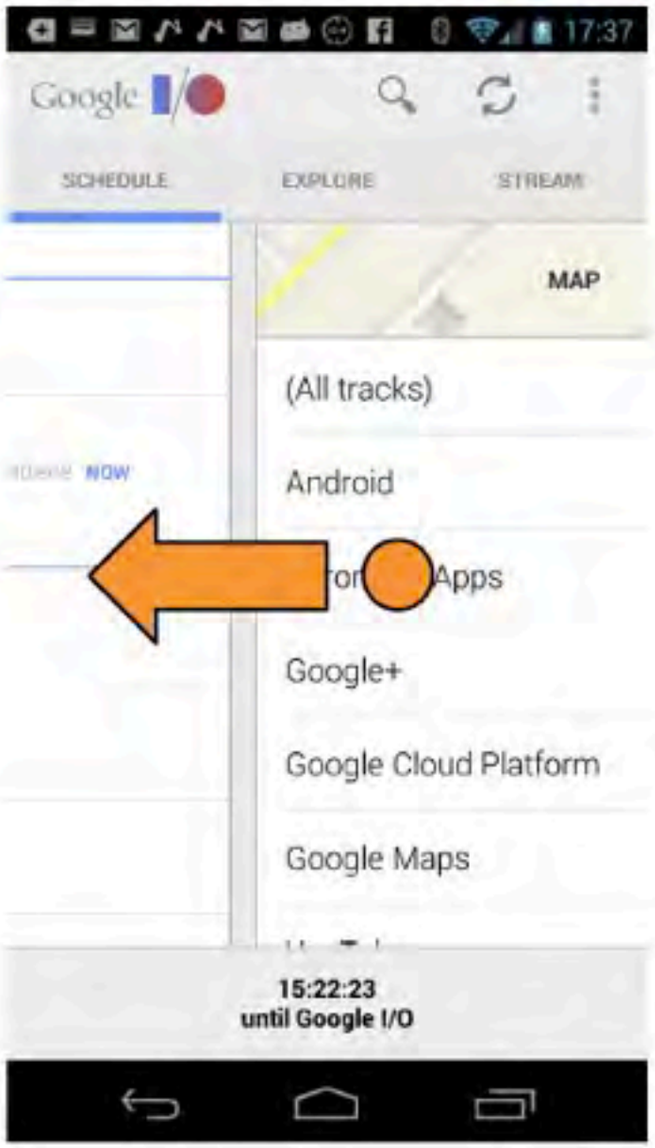
Fixed tabs

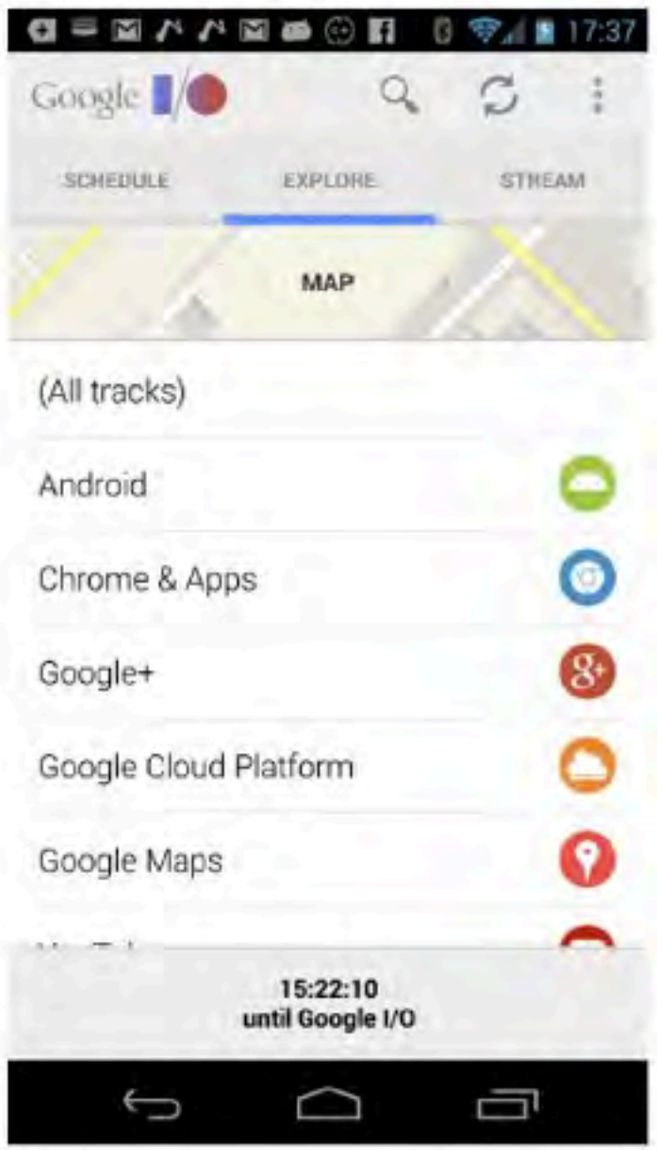

17:33
Google I/O
SCHEDULE
EXPLORE
STREAM
TUESDAY, MAY 14
14:00
On-site check-in
Level 1 NOW
16:00
Ingress game
Level 1, for early I/O attendees NOW
WEDNESDAY, MAY 15
07:00
Breakfast
Level 1
07:00
On-site check-in
Level 1
09:00
Keynote
Level 3 LIVE
15:26:17
until Google I/O



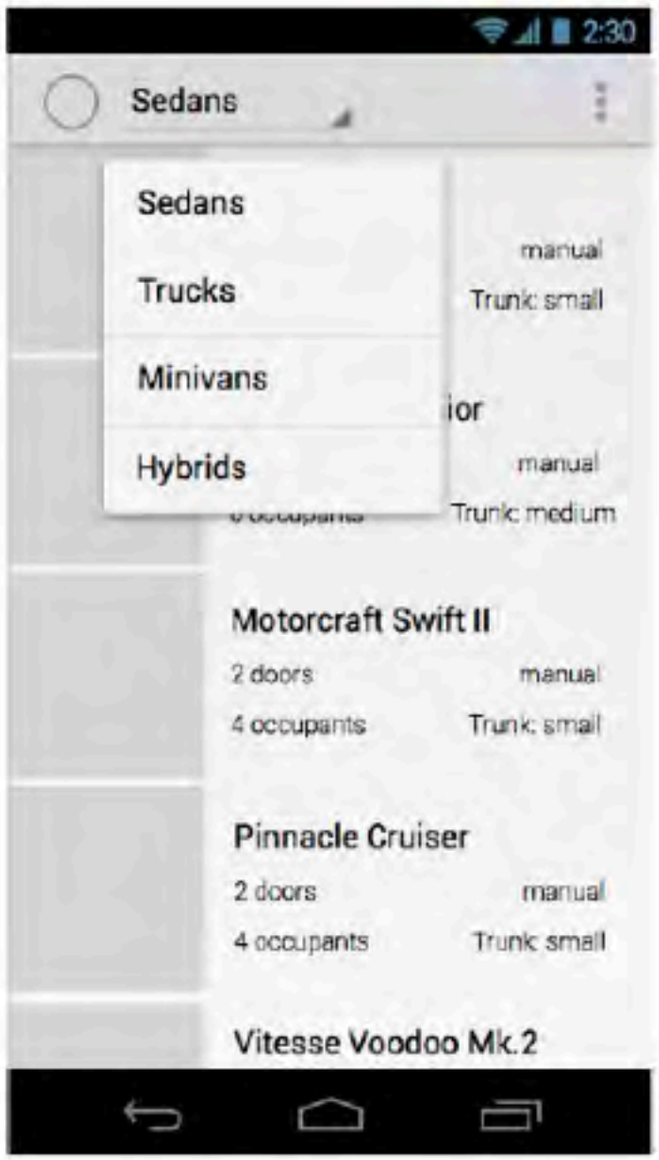
Fixed tabs: support side swipe

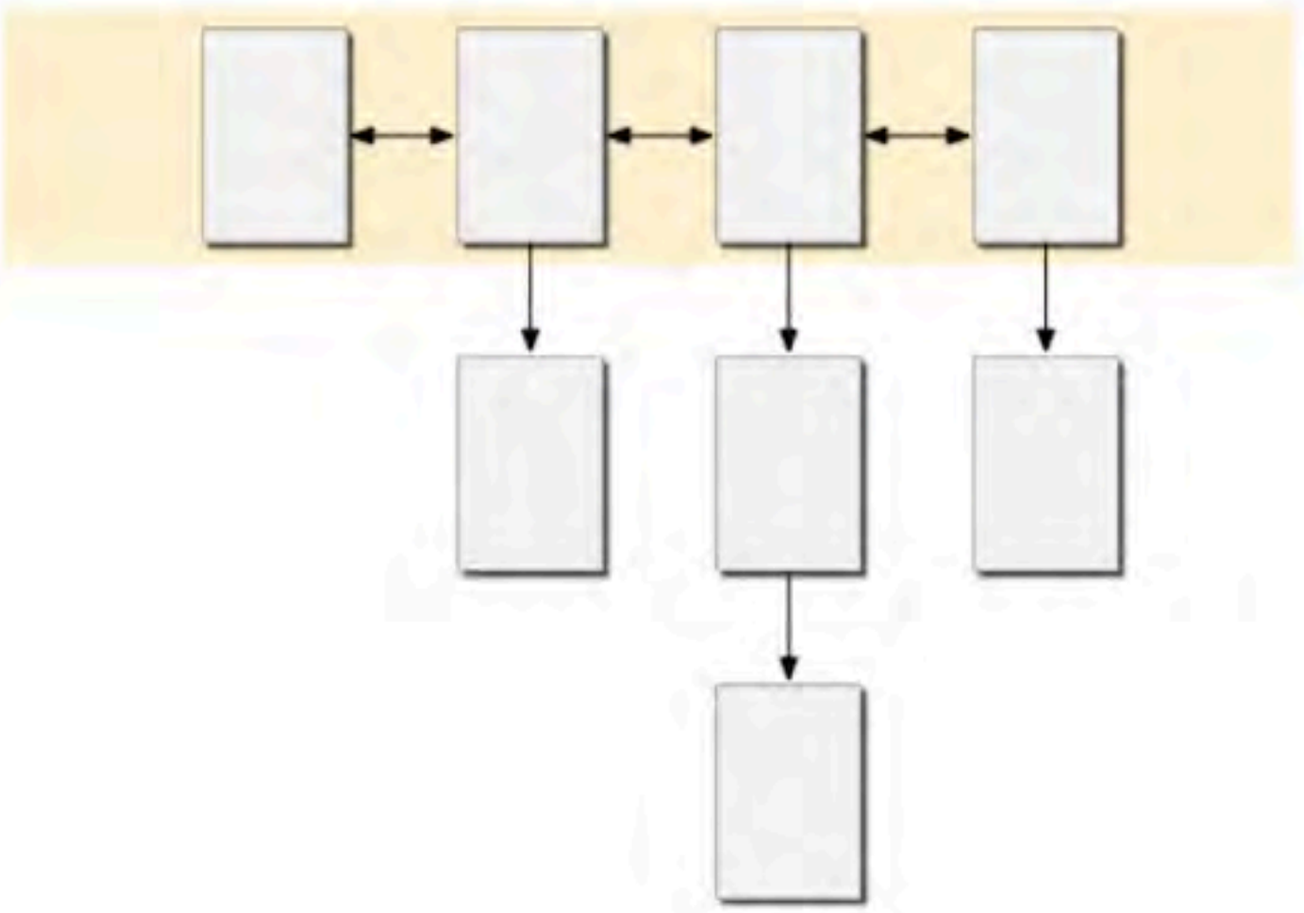




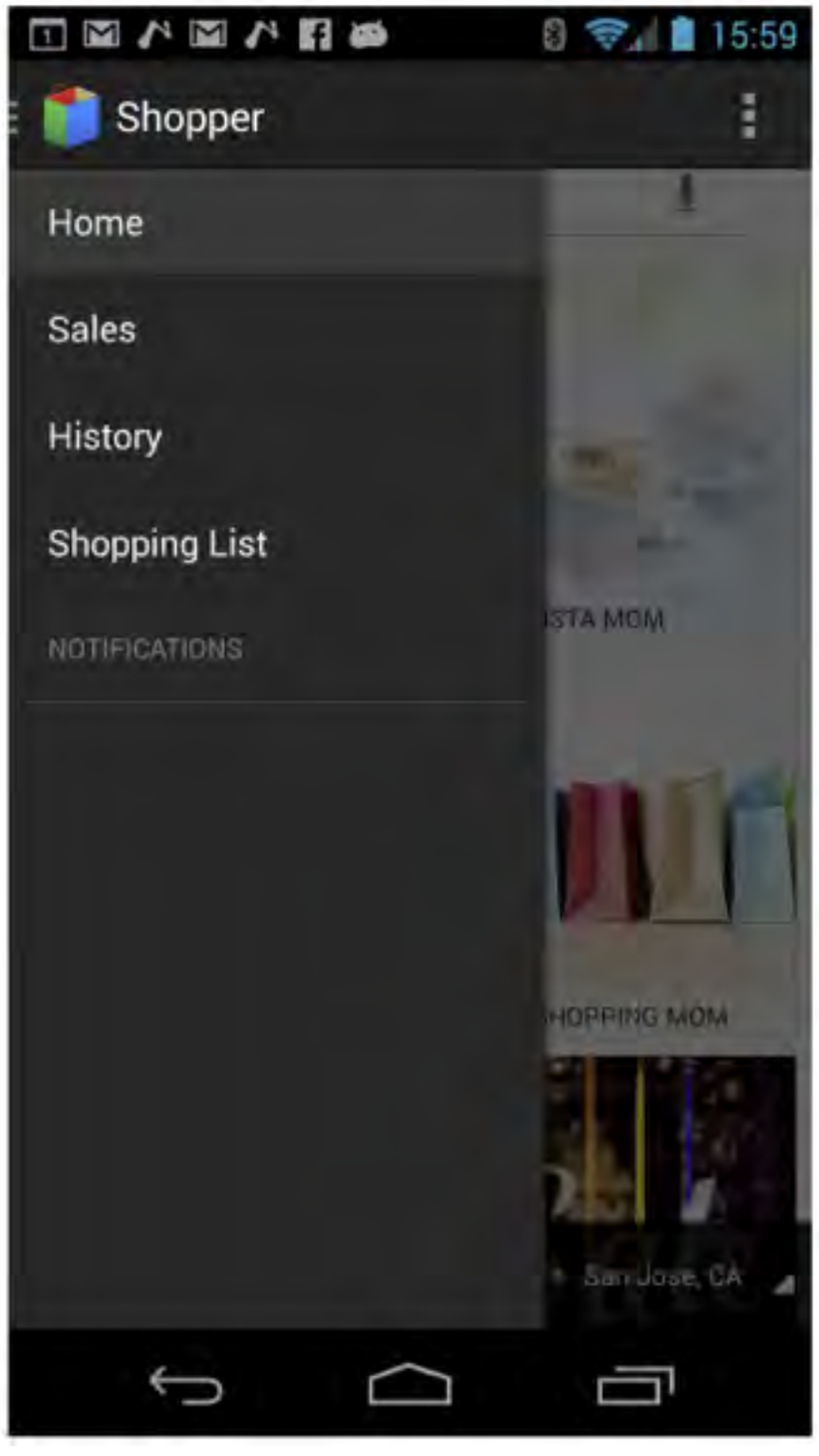


Spinners

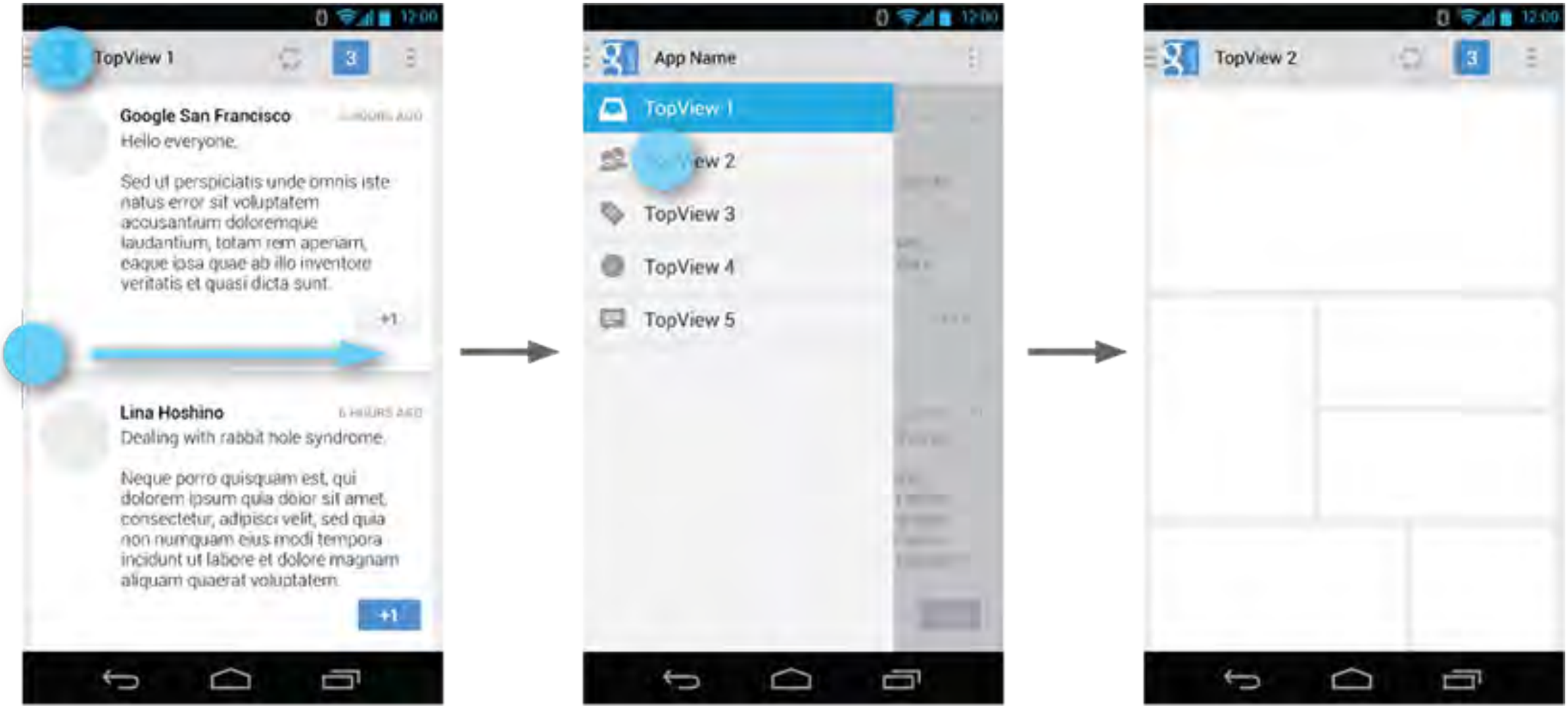

2:30
Sedans
Sedans
Trucks
Minivans
Hybrids
manual
Trunk: small
ior
manual
Trunk: medium
Motorcraft Swift II
2 doors
manual
4 occupants
Trunk: small
Pinnacle Cruiser
2 doors
manual
4 occupants
Trunk: small
Vitesse Voodoo Mk.2



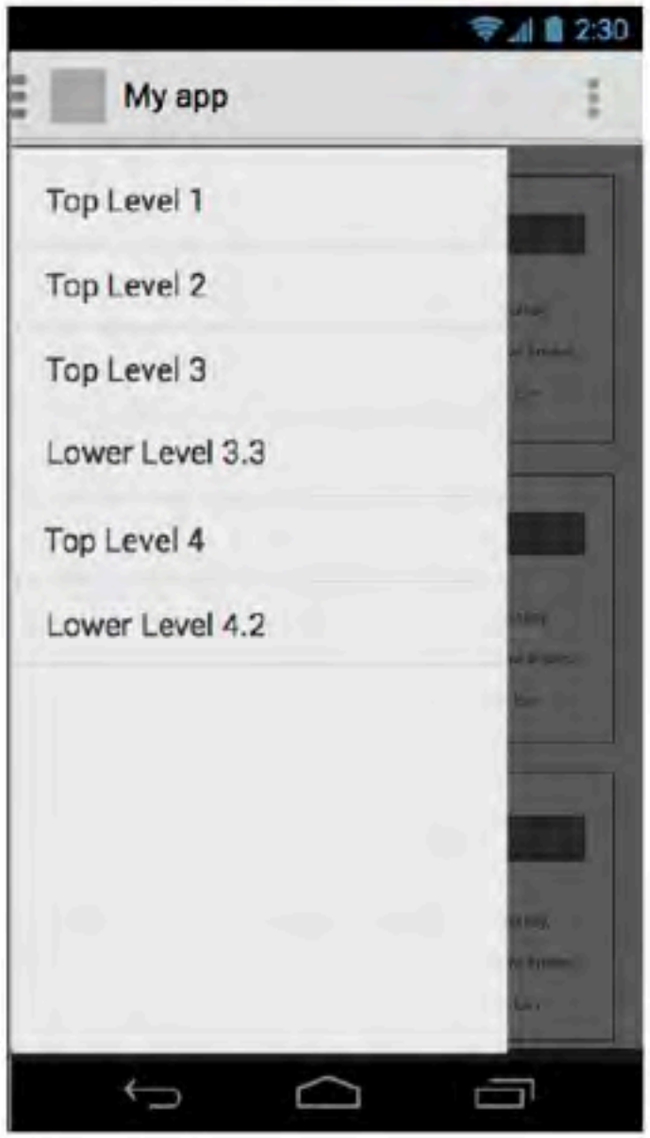
Navigation Drawer

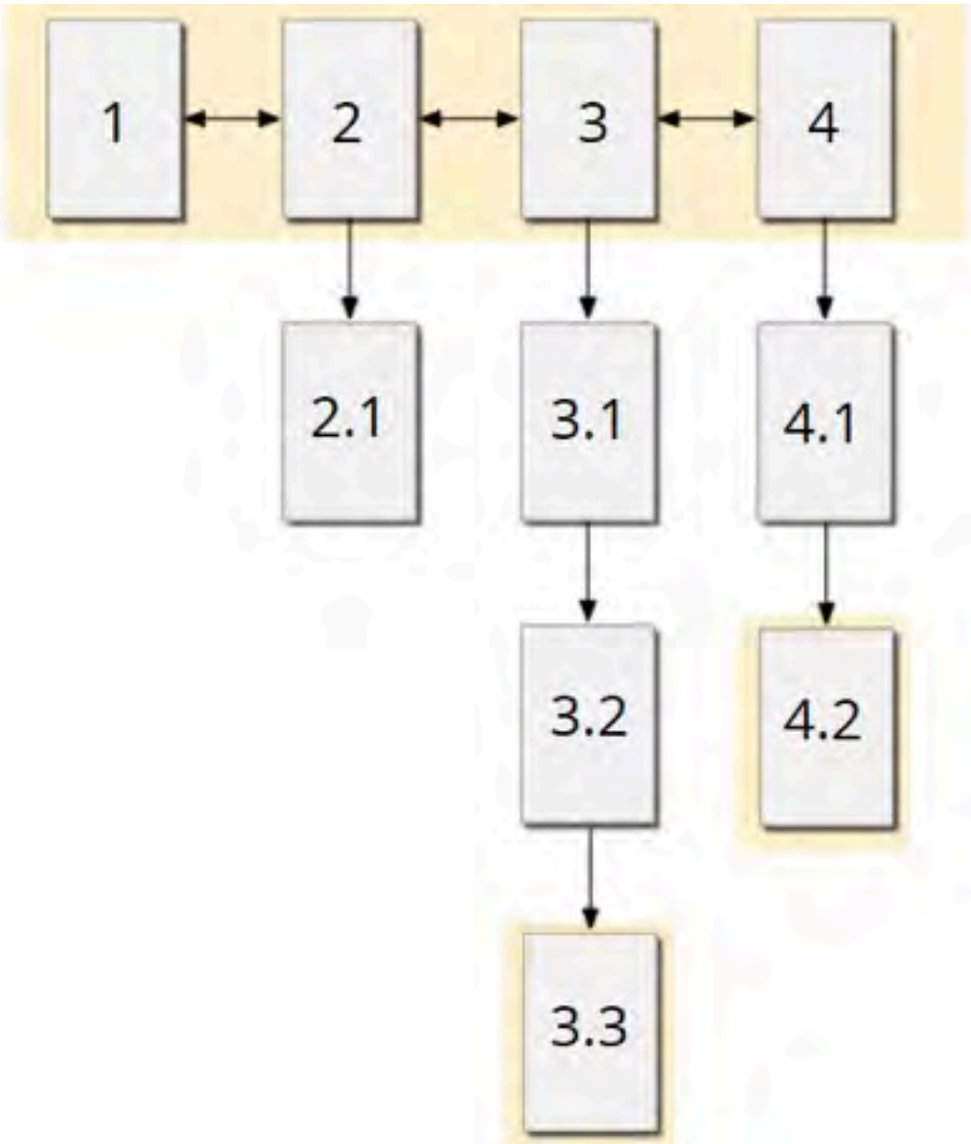


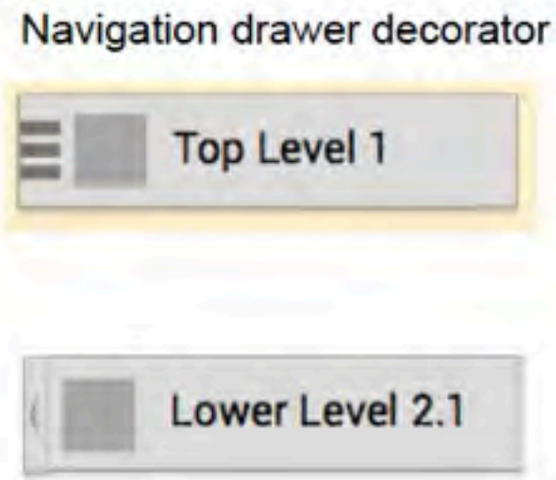
Navigation Drawer



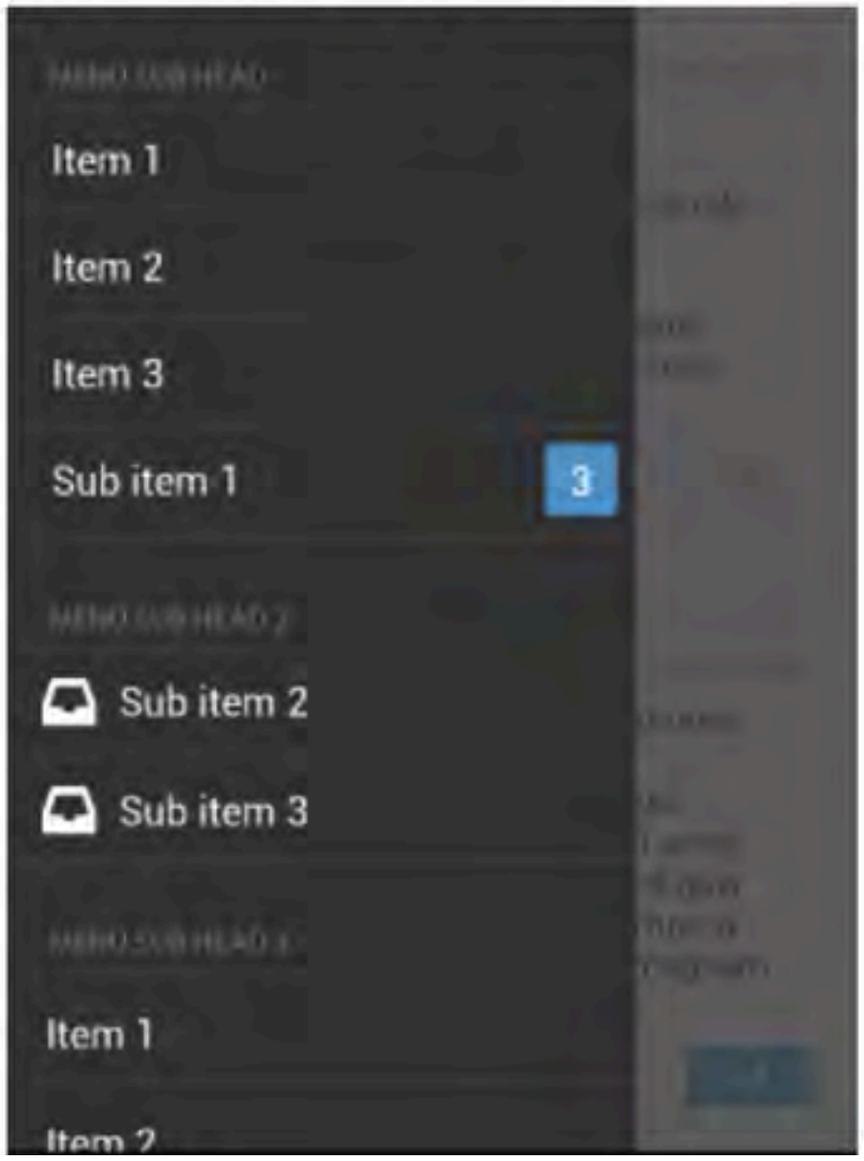
Navigation hubs



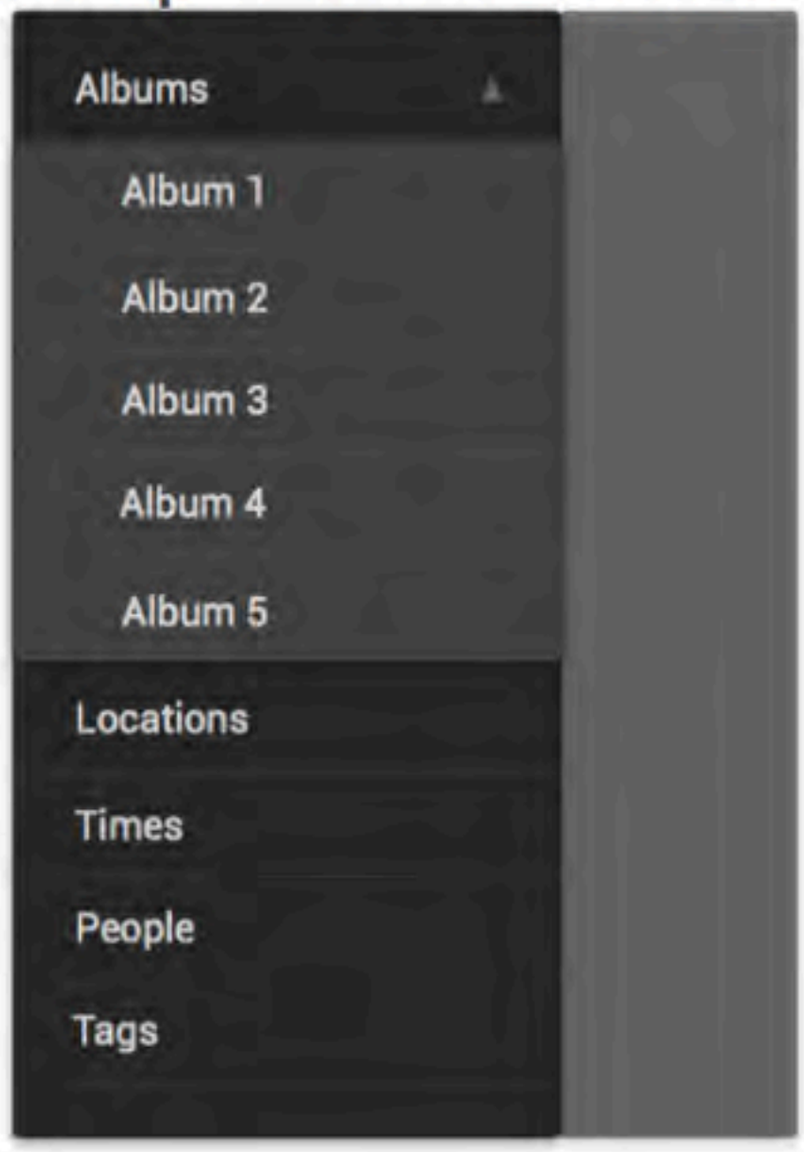




Dividers, icons, counters



Collapsible items



Assignment #1

- Design a simple (native) mobile application

Step 1: View this video online

- Structure in Android App Design

- https://developers.google.com/events/io/sessions/326301704

Class List App

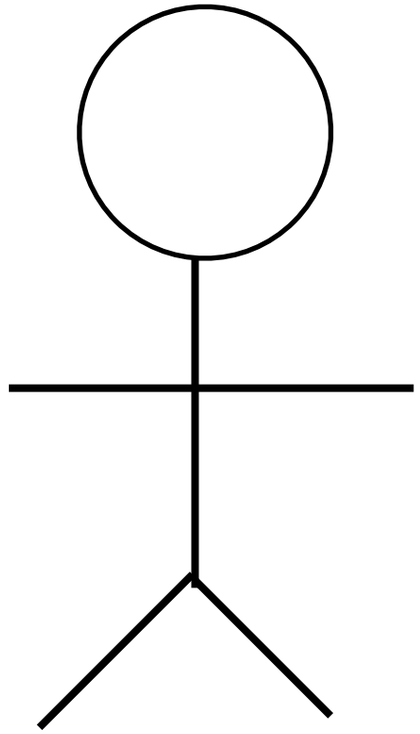
- **CS 150 Class List app**
- An Android app that when launched lists the students and teachers in the class
- The app allows the user to browse filtered lists of all members of the class, students & teachers
- The app allows the user to view a detailed screen of a member of the class including a photo, name, email and phone number

Class List App

- **Logistics**
- To start, we will work on the design
- Submit your work in whatever format you choose: can be PPT or Word document, or hand sketches

A. List the actors (users)

- Users should be the end user, not the creators of content



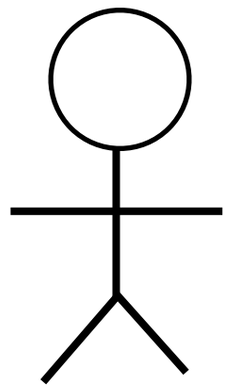
User

B. Write the use cases

- Limit to the top 3-5 use cases
  - View all class members in a list
  - View teachers in a list
  - View students in a list
  - Look at the details for a person

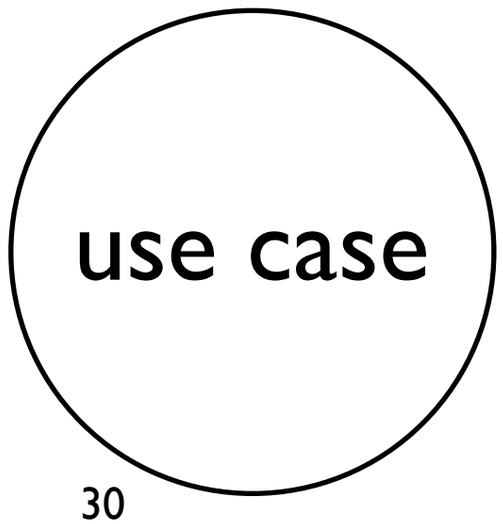
C. Create the use case diagram

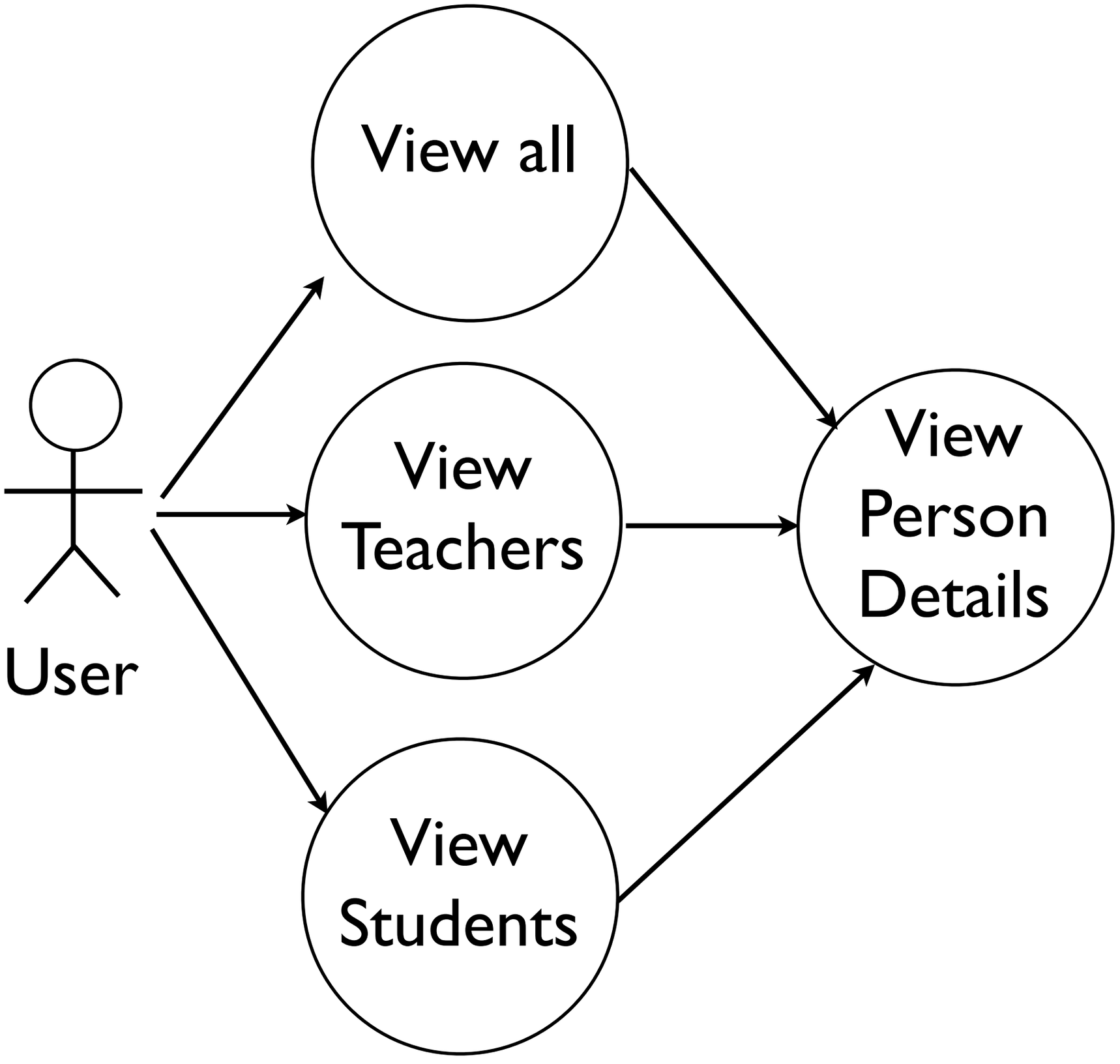
- Include
  - actor(s)



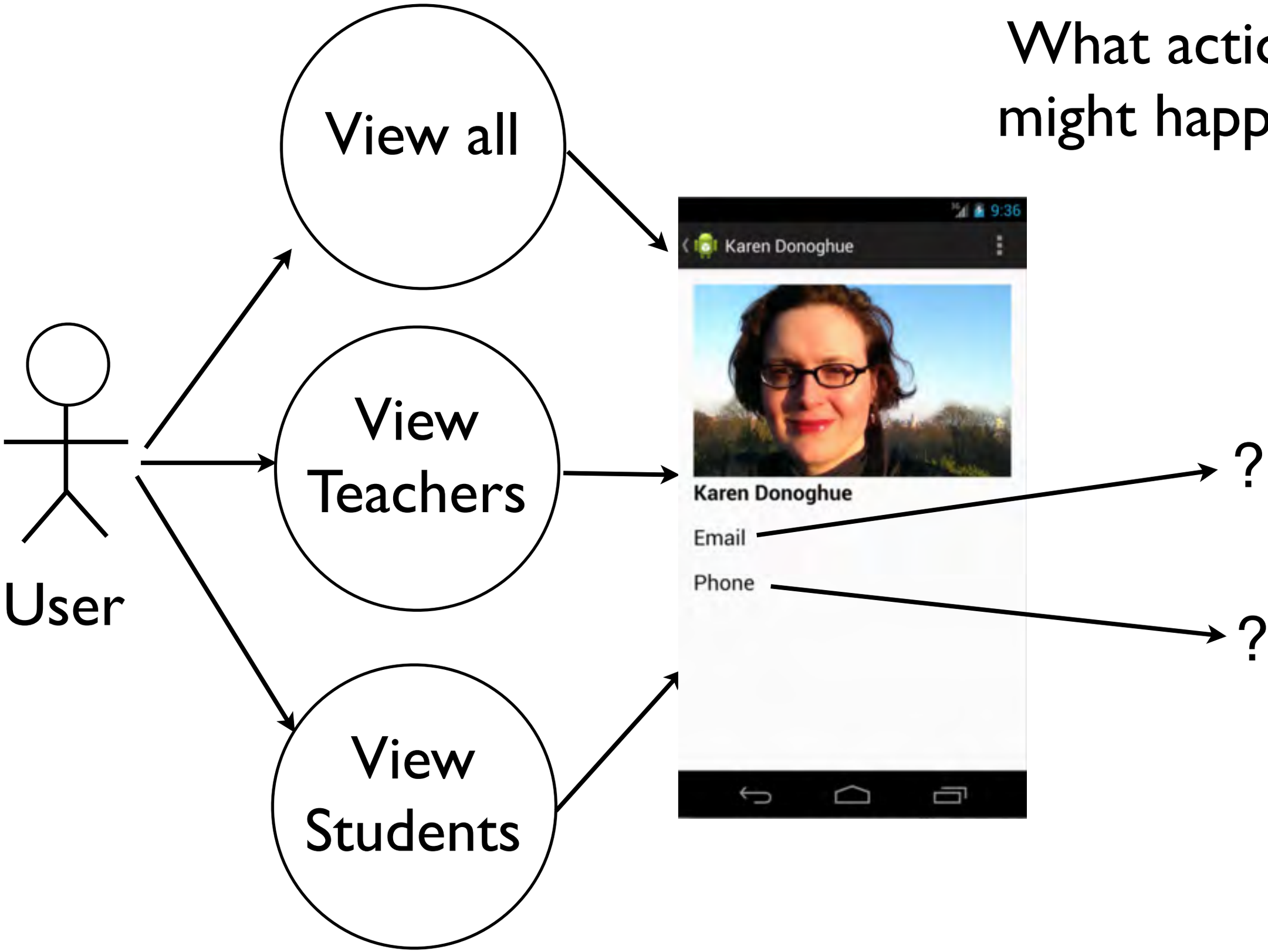
  - relationships

  - use cases



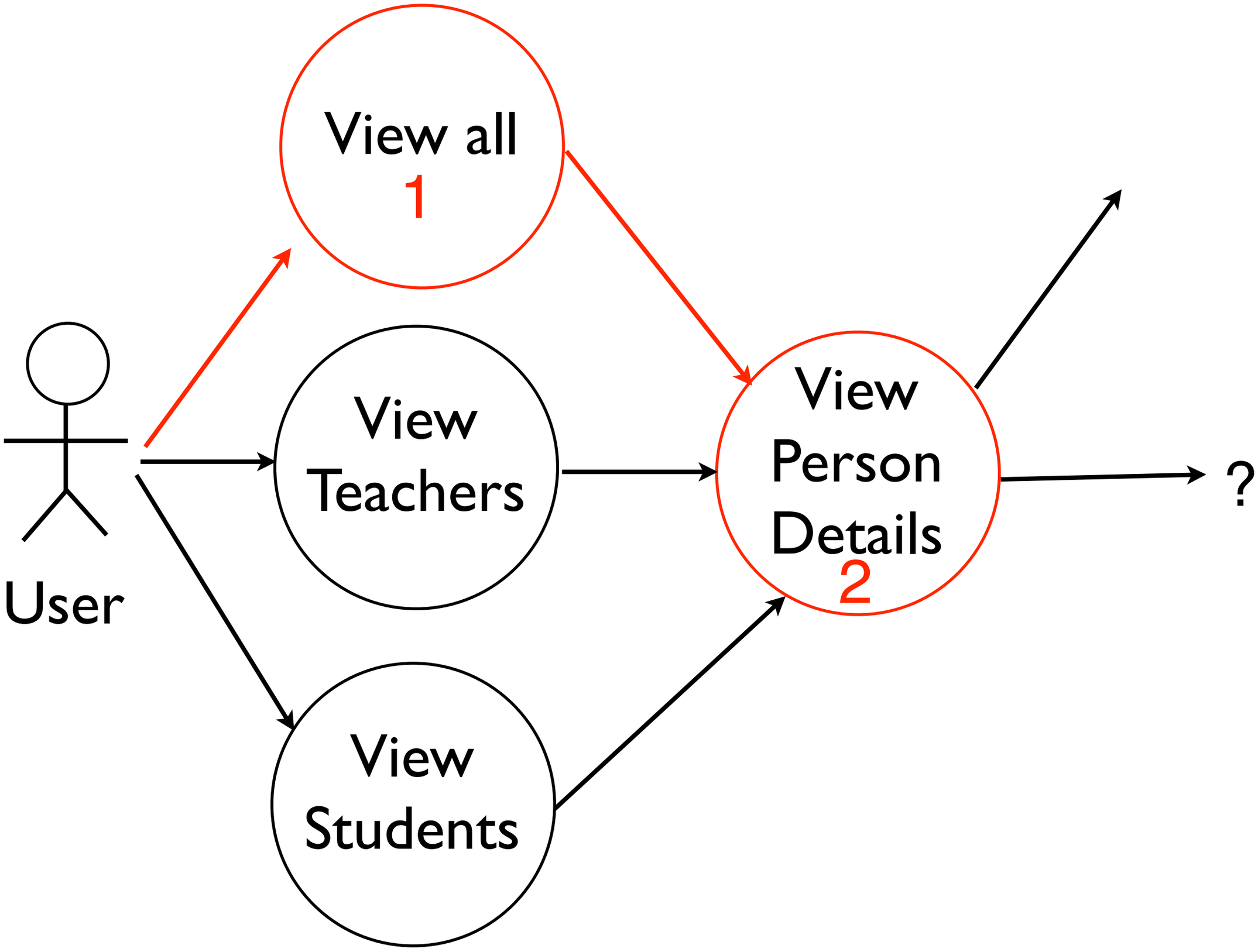

View all
View Teachers
View Person Details
User
View Students

What actions might happen?

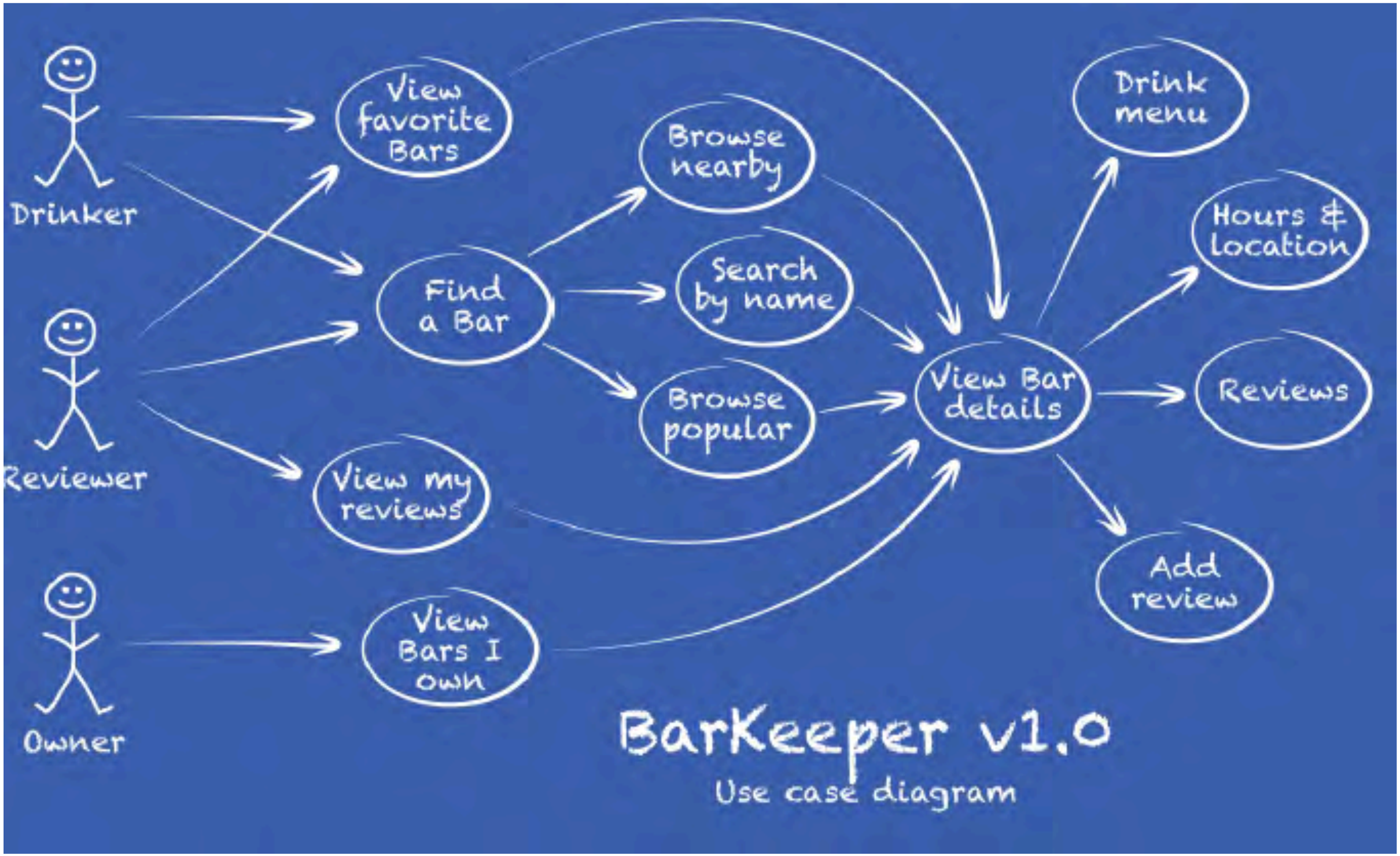


D. List the use cases in sequence

- Define the sequence in which the top 3-5 use cases occur


View all
1
View
Teachers
View
Person
Details
2
?
User
View
Students

Why does sequence of use cases matter?



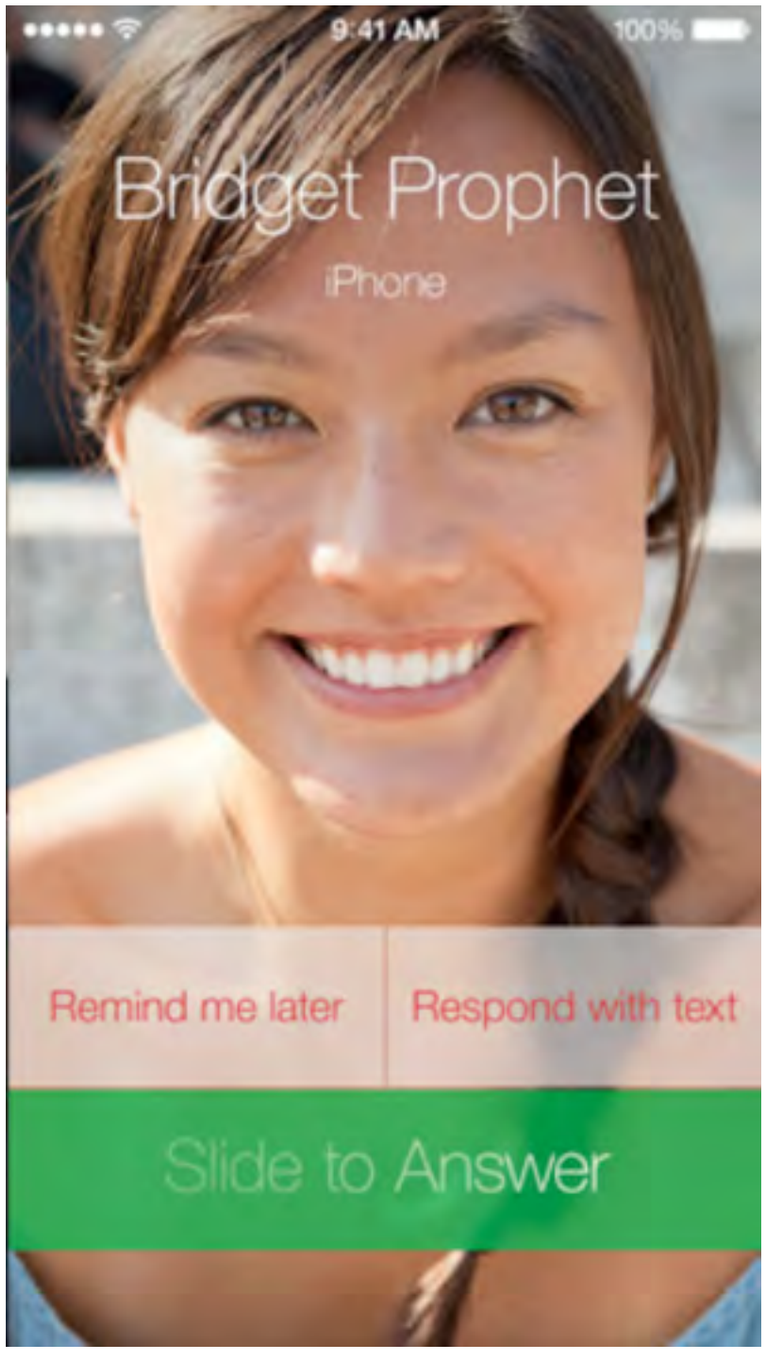
Reproduced from work created and [shared by the Android Open Source Project](#) and used according to terms described in the [Creative Commons 2.5 Attribution License](#).

E. Decomposition

- List what are the most frequently used pathways through your app?
  - View all ⟶ View Person details
  - View Teachers ⟶ View Person details

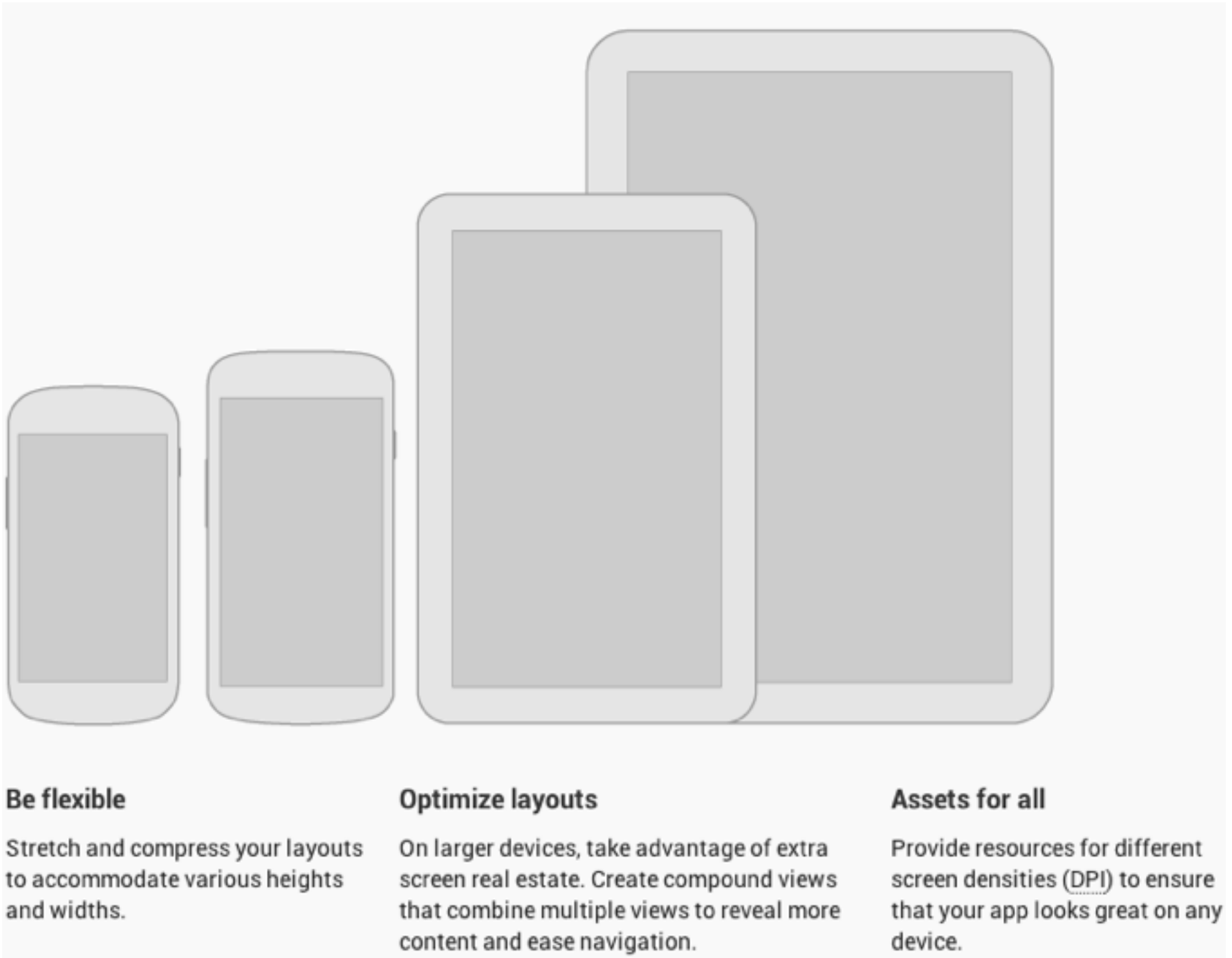
F. What happens when...?

- What should your app do if the user receives a phone call while using it?



F. What happens when...?

- What should your app do if the user rotates the phone?
- What if the user tries to view your app on a larger screen or other device like a table?



Be flexible

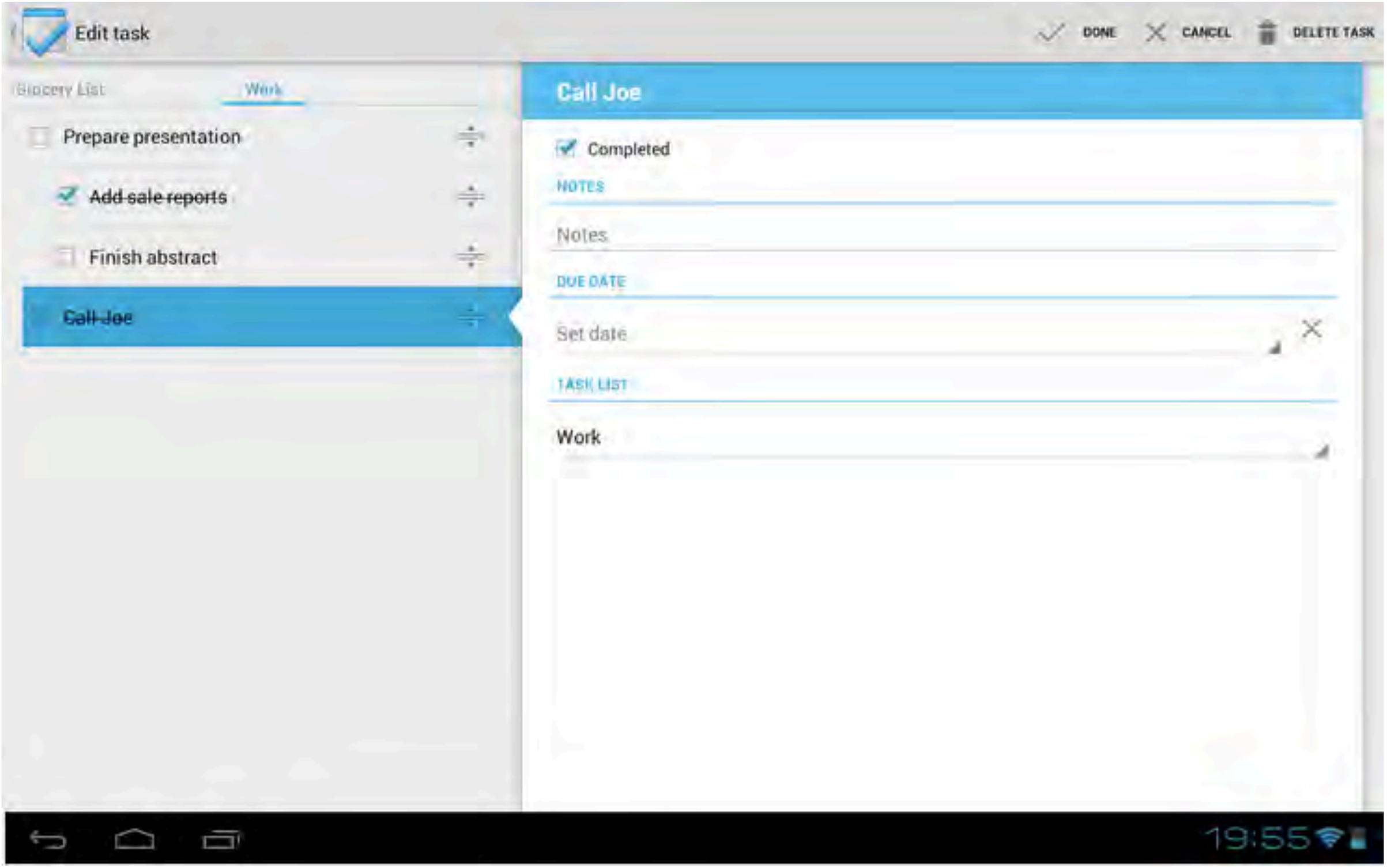
Stretch and compress your layouts to accommodate various heights and widths.

Optimize layouts

On larger devices, take advantage of extra screen real estate. Create compound views that combine multiple views to reveal more content and ease navigation.

Assets for all

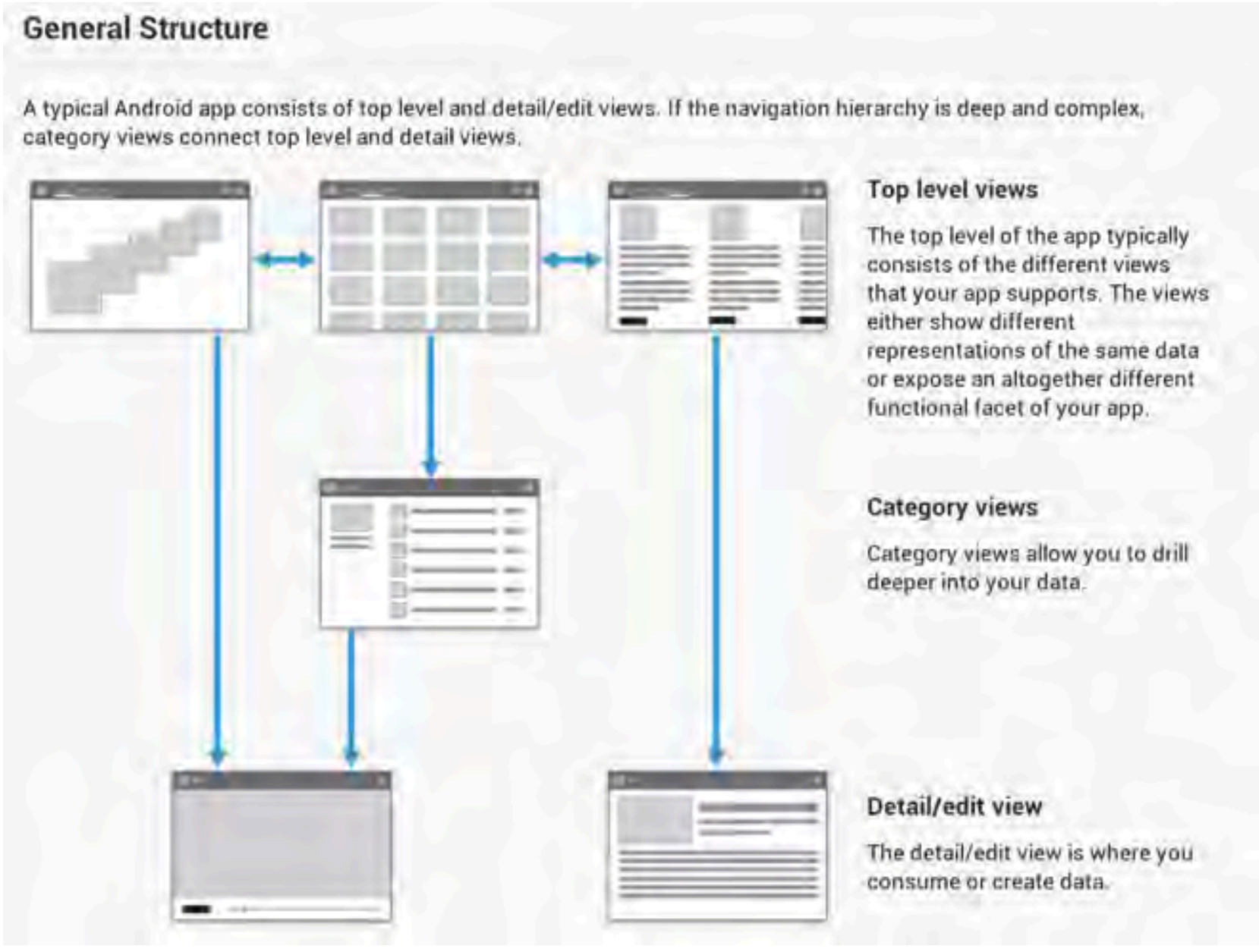
Provide resources for different screen densities (DPI) to ensure that your app looks great on any device.



https://play.google.com/store/apps/details?id=ch.teamtasks.tasks.paid&hl=en

G. Define the app hierachy

- For the major use cases, map each use case in the diagram to the application categories of data (refer to diagram at right for application categories)
- For example, the topmost category, the detailed category, etc



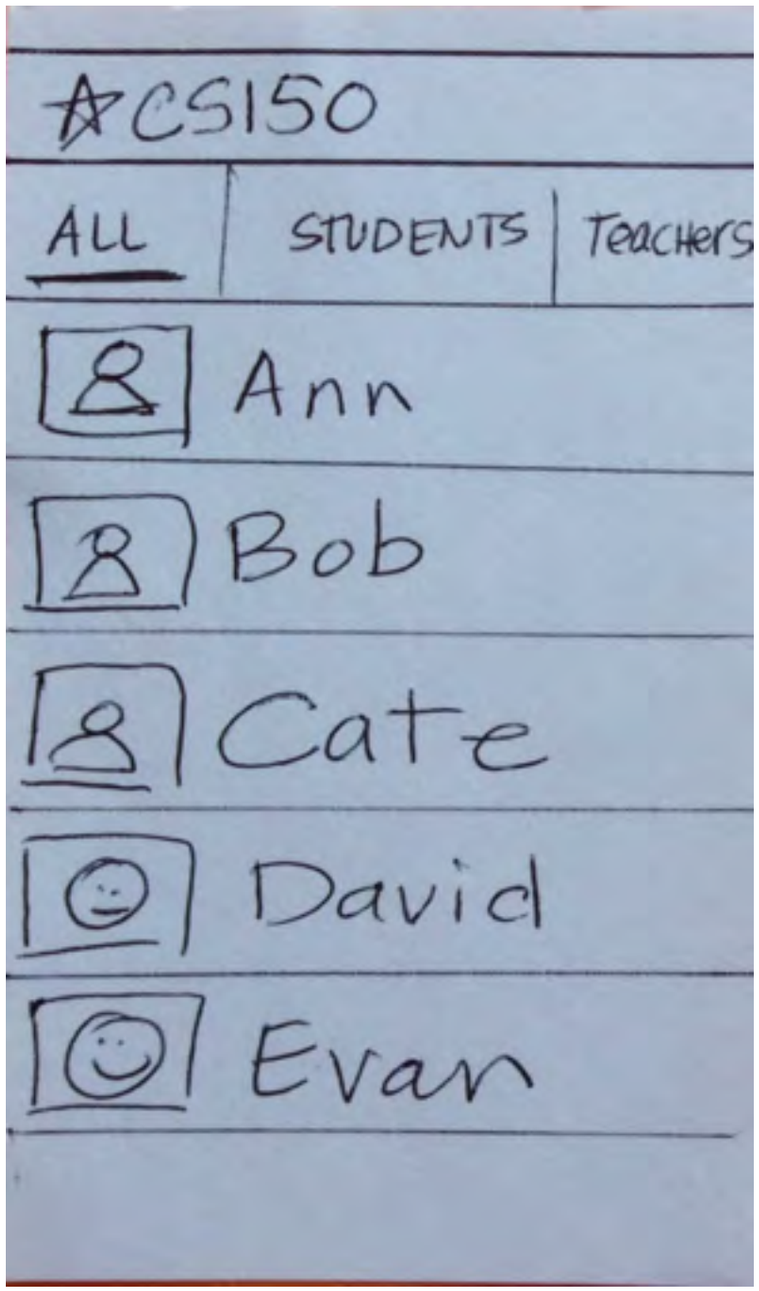
H. Define the major screens

- For the major use cases, map each use case in the diagram to an application screen and the UI components that are needed to support the use case

View all

View Teachers

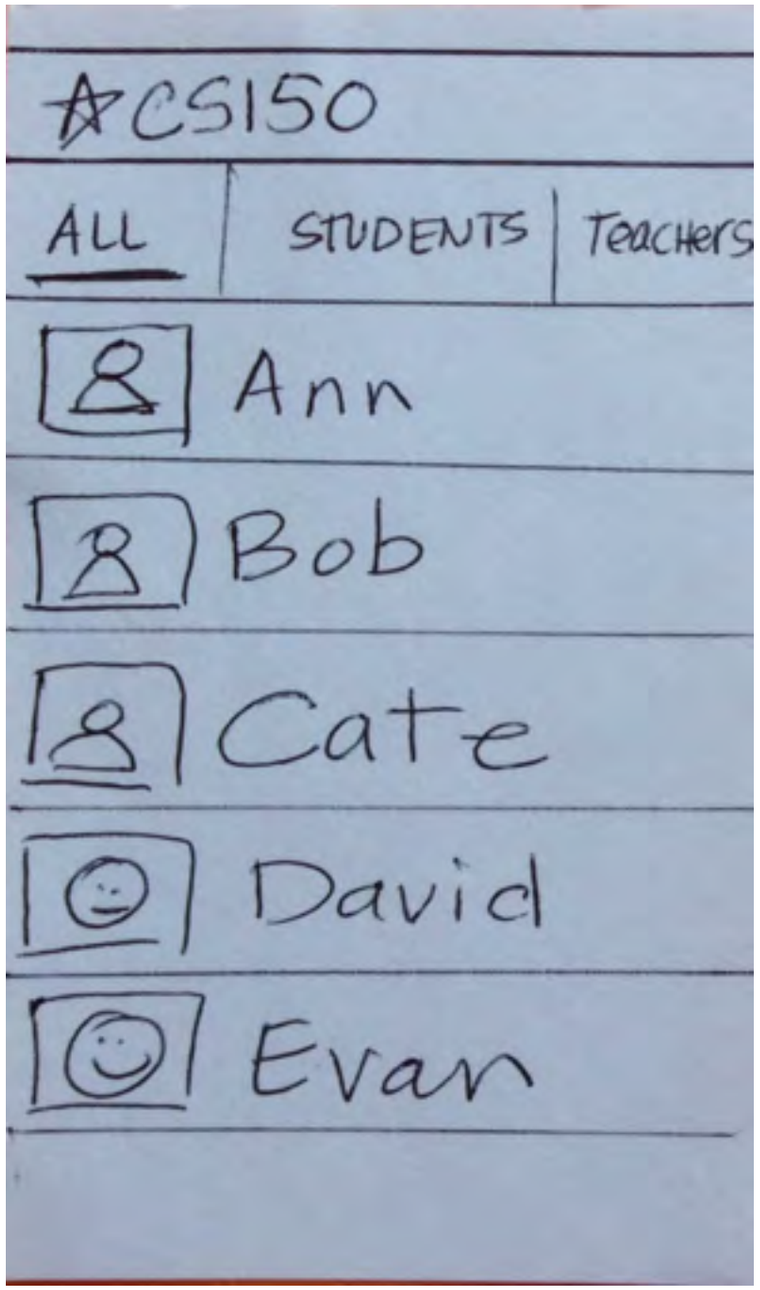
View Students

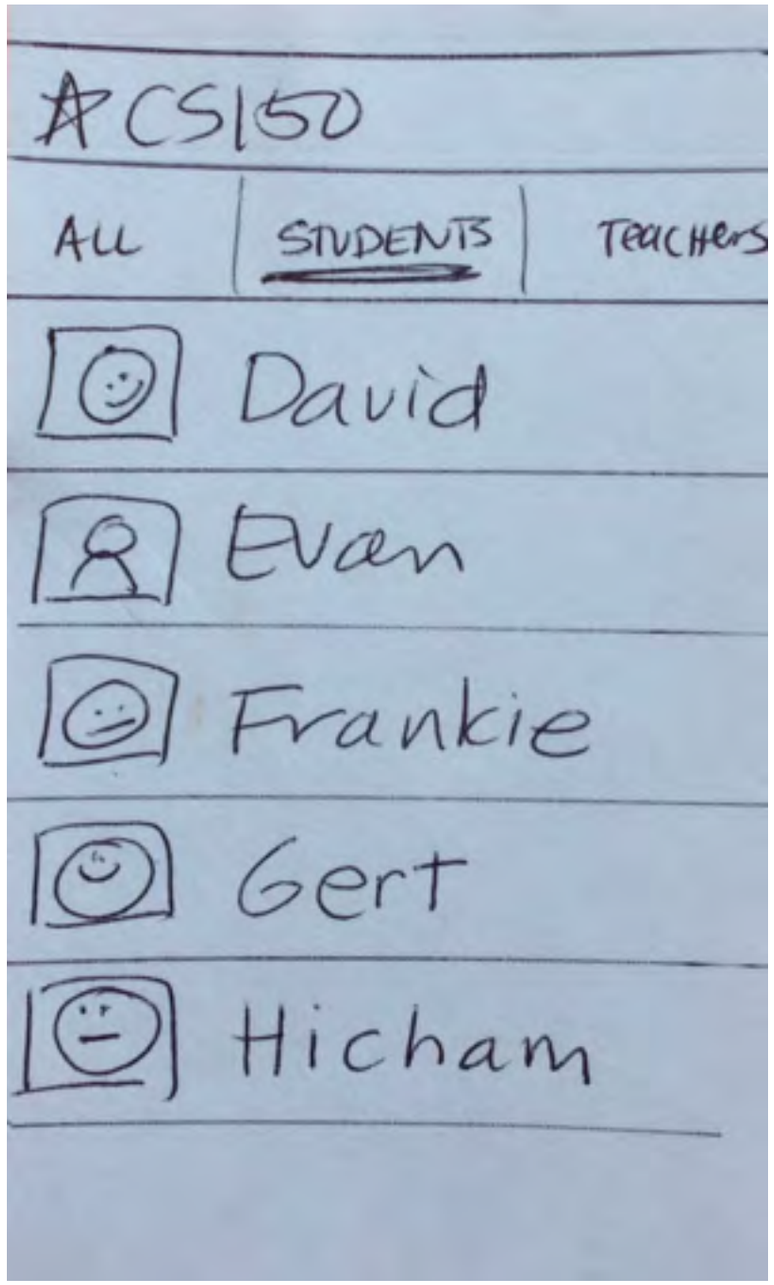


View all

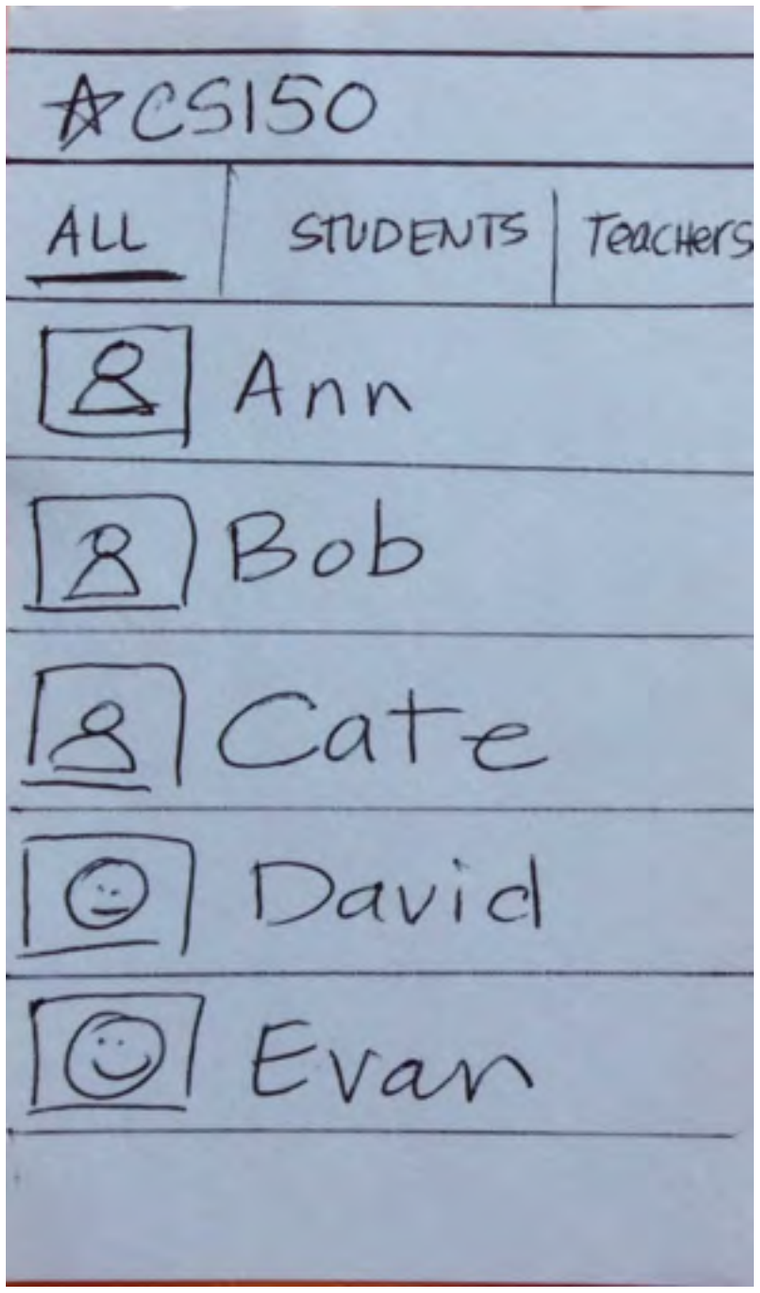
View Students

View Teachers

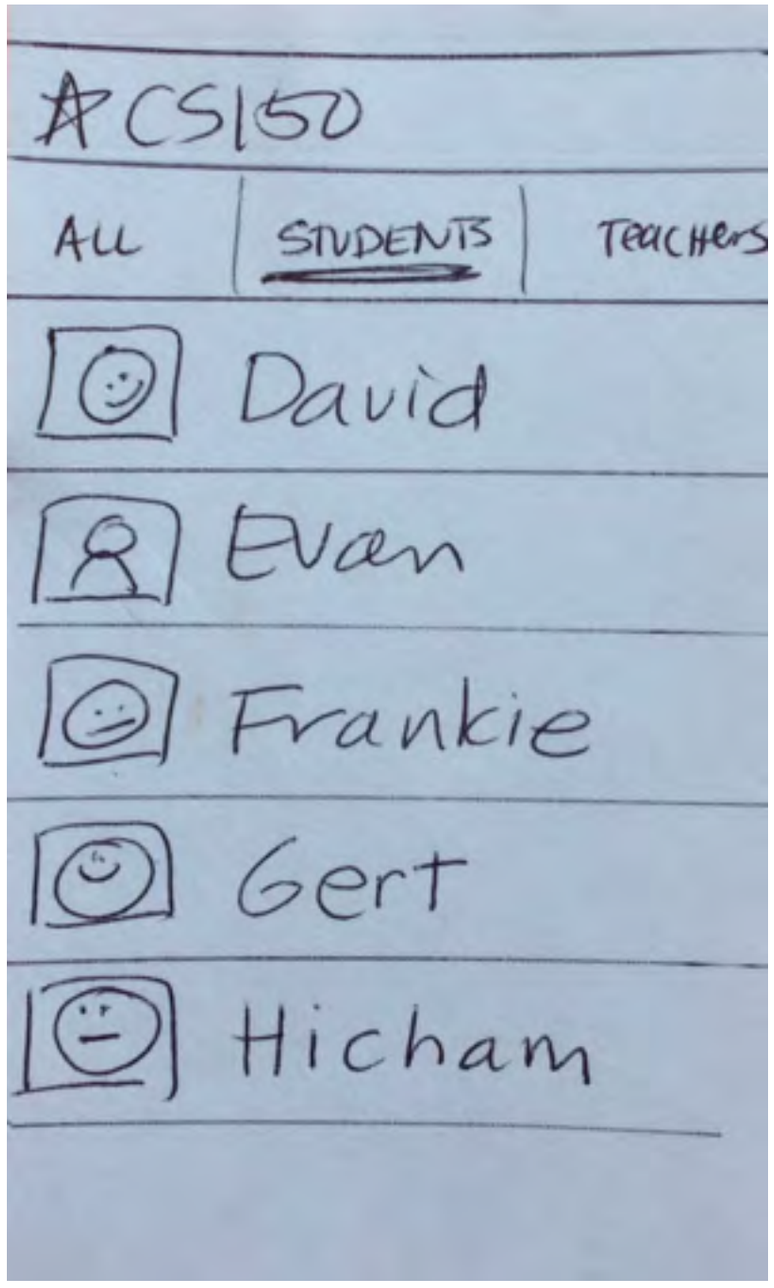




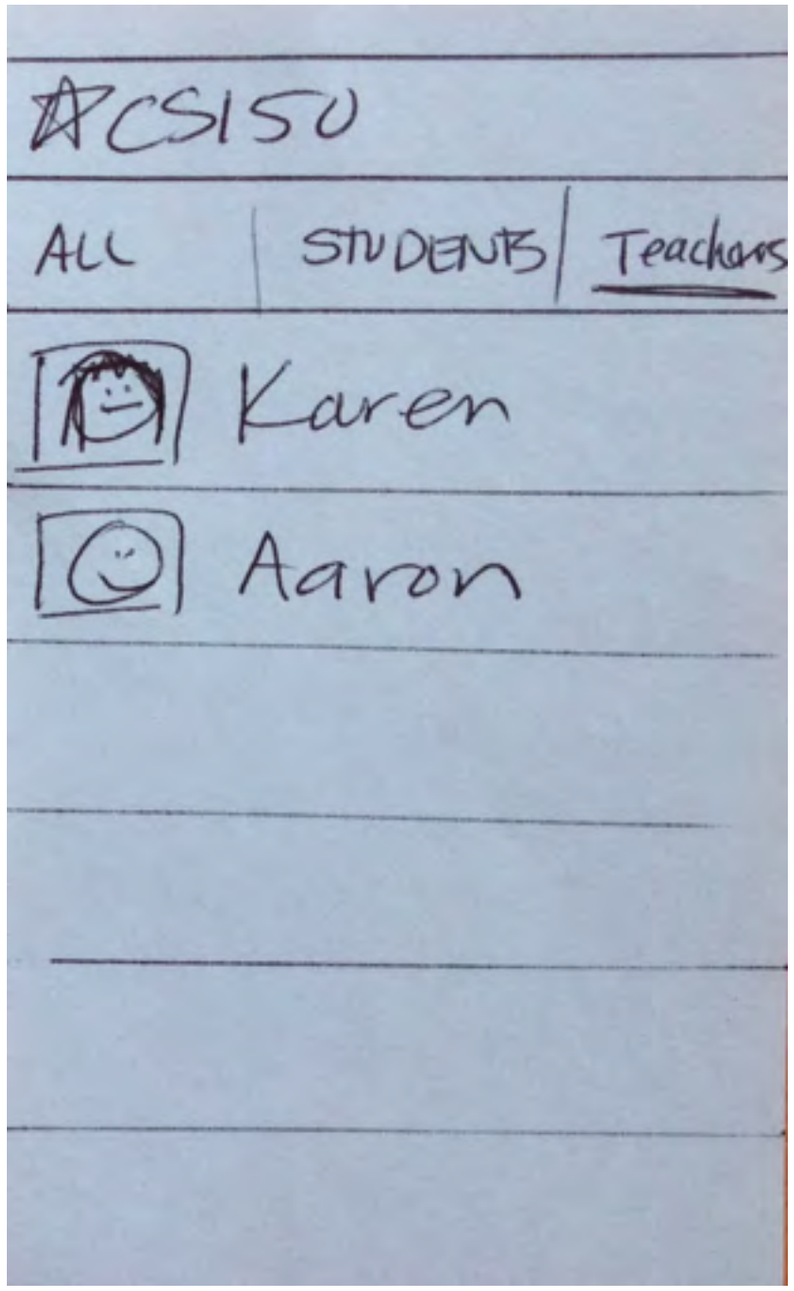
View all



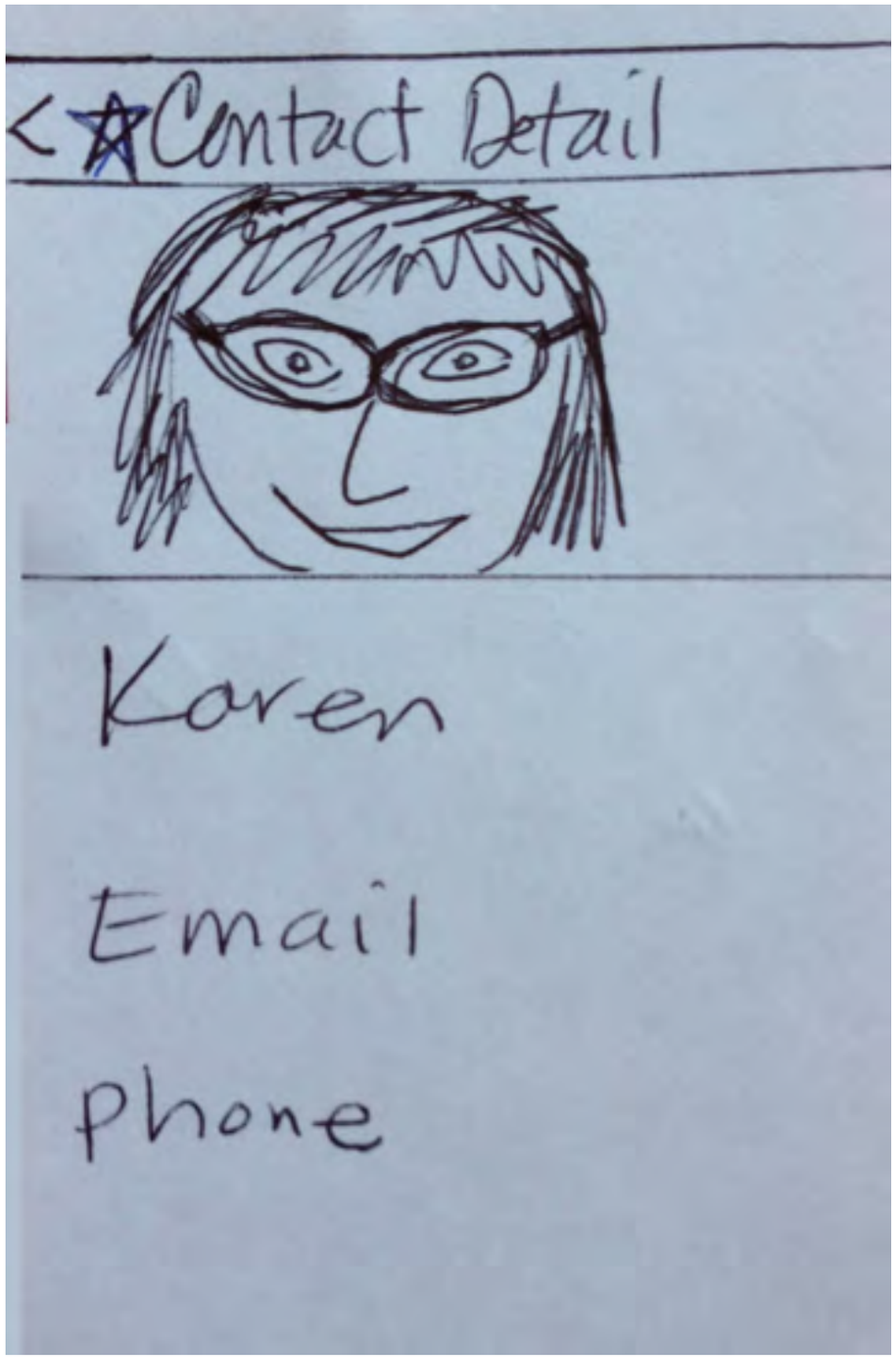
View Teachers



View Students

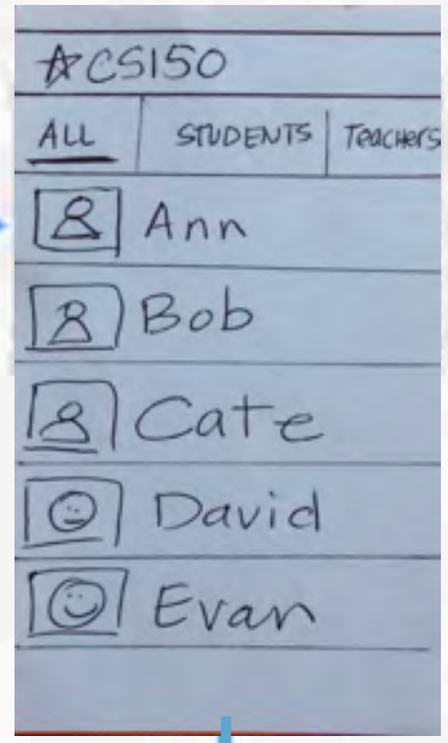


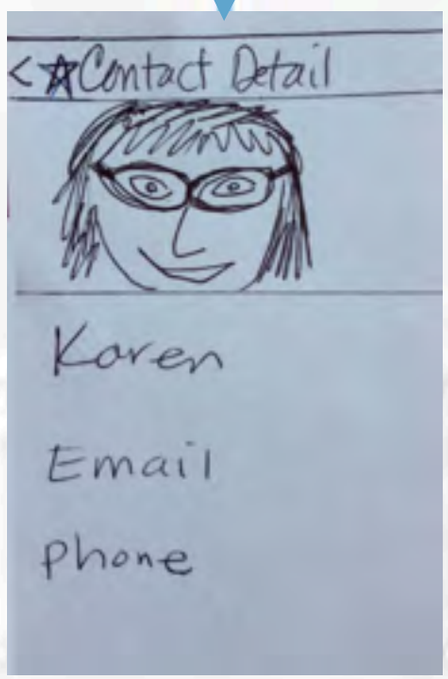
View Person Details



General Structure

A typical Android app consists of top level and detail/edit views. If the navigation hierarchy is deep and complex, category views connect top level and detail views.





Top level views

The top level of the app typically consists of the different views that your app supports. The views either show different representations of the same data or expose an altogether different functional facet of your app.

Category views

Category views allow you to drill deeper into your data.

Detail/edit view

The detail/edit view is where you consume or create data.

Questions?

- Ready for Assignment 2?

Assignment #2

- Build your ClassList application as a native mobile app using Android Studio

Build a working app

- Build the application to demonstrate the top 3-5 use cases
- Be able to demonstrate the app in class either in the emulator or on a device

Android Design

- Visit http://developer.android.com/design/index.html
- This is a good reference for UI design and assets

Class List App

- **Logistics**
- We will use Android Studio to build the app
  - Please install the tool Android Studio: http://developer.android.com/sdk/installing/studio.html
  - If you have an Android handset, make sure it is Debugging enabled
  - Go to Settings and make sure you have Developer Options as a choice. Set USB Debugging to ON
    - for Nexus 4 do this: http://irwinj.blogspot.com/2013/02/setting-up-android-debugging-on-nexus-4.html
  - Get the sample data (Names, Photos) and use this in your app in the ZIP file **Assignment2_People.zip**

Android Development

- Visit http://developer.android.com/develop/index.html
- This is a good reference for developing

Get Started
Style
Devices and Displays
Themes
Touch Feedback
Metrics and Grids
Typography
Color
Iconography
Writing Style
Patterns
Building Blocks
Downloads
Videos

Metrics and Grids

PREVIOUS NEXT

Devices vary not only in physical size, but also in screen density (DPI). To simplify the way you design for multiple screens, think of each device as falling into a particular size bucket and density bucket:

- The size buckets are *handset* (smaller than 600dp) and *tablet* (larger than or equal 600dp).
- The density buckets are LDPI, MDPI, HDPI, XHDPI, and XXHDPI.

Optimize your application's UI by designing alternative layouts for some of the different size buckets, and provide alternative bitmap images for different density buckets.

Because it's important that you design and implement your layouts for multiple densities, the guidelines below and throught the documentation refer to layout dimensions with dp measurements instead of pixels.

Space considerations

Devices vary in the amount of density-independent pixels (dp) they can display.

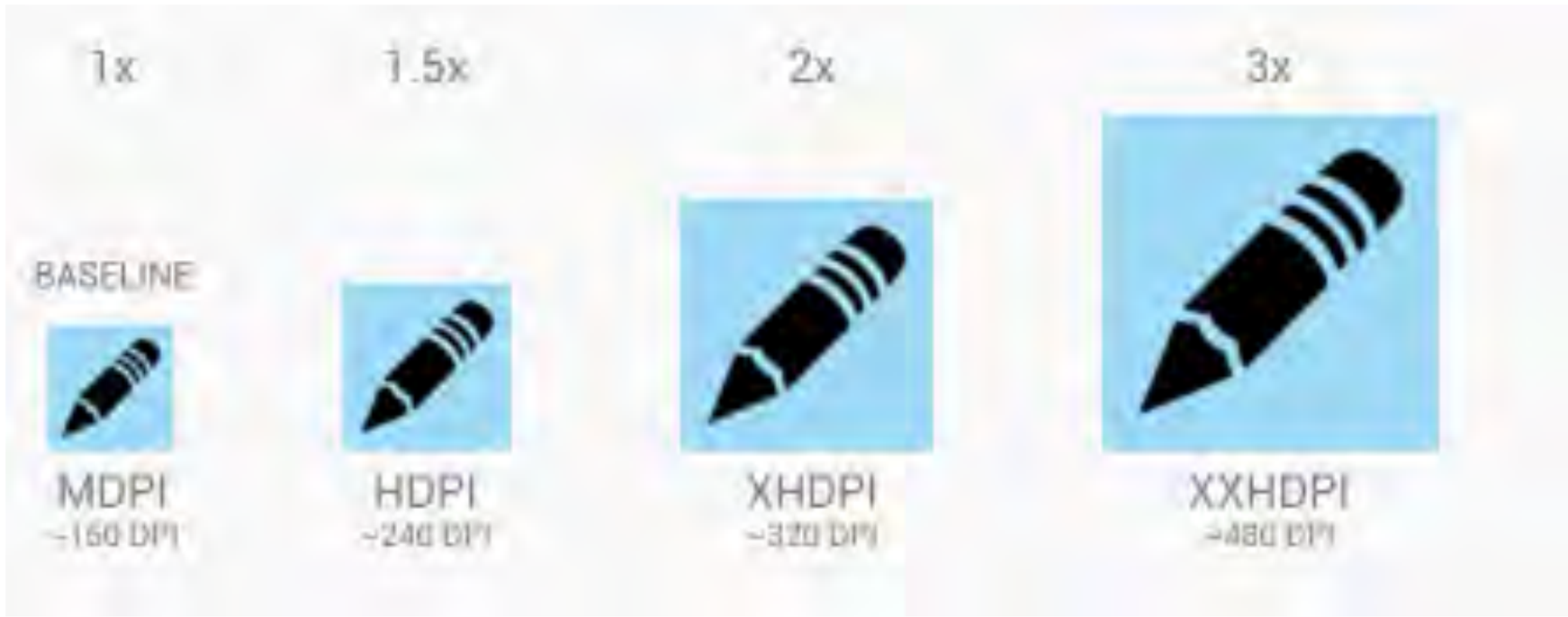
To see more, visit the Screen Sizes and Densities Device Dashboard.

Thumbnails and images

- Android has multiple screen sizes so visual assets are delivered in several resolutions
- In the sample data we will use XHDPI

Density-independent pixel (dp)

- A virtual pixel unit that you should use when defining UI layout, to express layout dimensions or position in a density-independent way.
- The density-independent pixel = one physical pixel on a 160 dpi screen. This is the baseline density assumed by the system for a "medium" density screen.



Density-independent pixel (dp)

- At runtime, the system transparently scales the dp units based on the actual density of the screen in use.
- The conversion of dp units to screen pixels is:
  - px = dp * (dpi / 160).
  - For example, on a 240 dpi screen, 1 dp = 1.5 physical pixels.
  - You should try aim to dp units when defining your application's UI, to ensure proper display of your UI on screens with different densities.

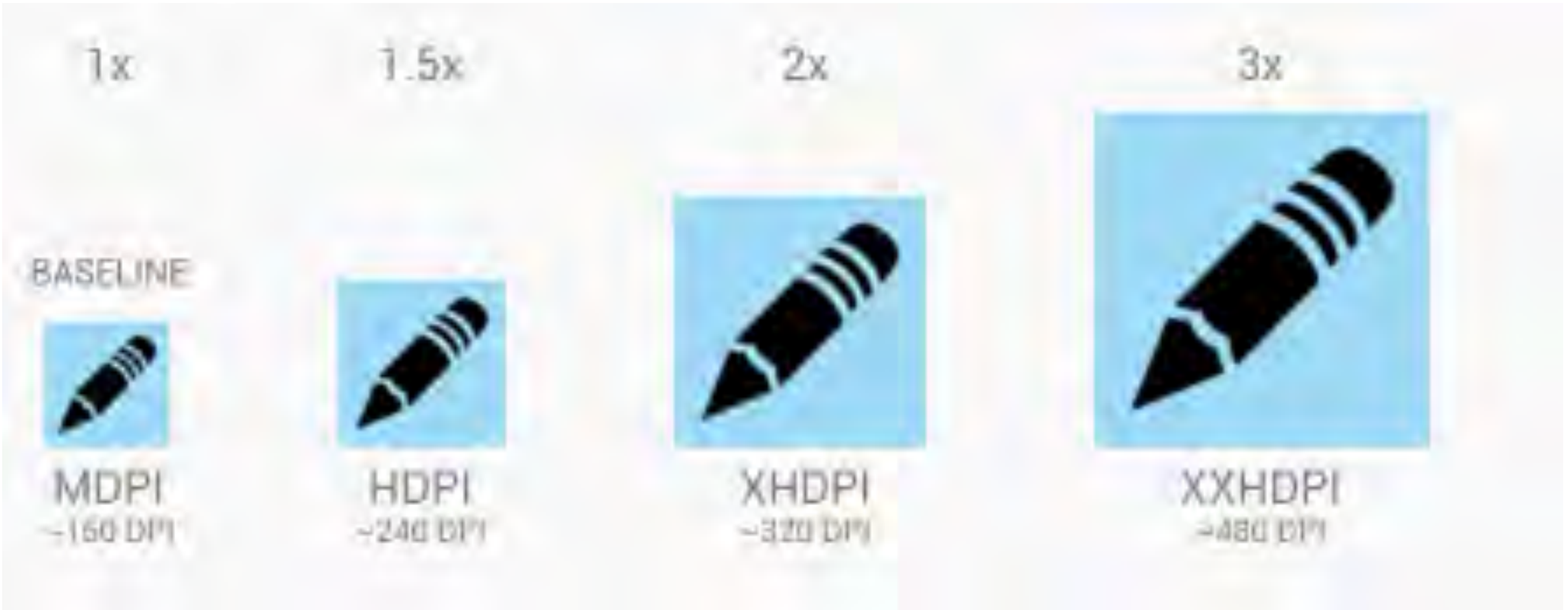
Design for varying display sizes

- The size buckets are *handset* (smaller than 600dp) and *tablet* (larger than or equal 600dp).
- The density buckets are LDPI, MDPI, HDPI, XHDPI, and XXHDPI.

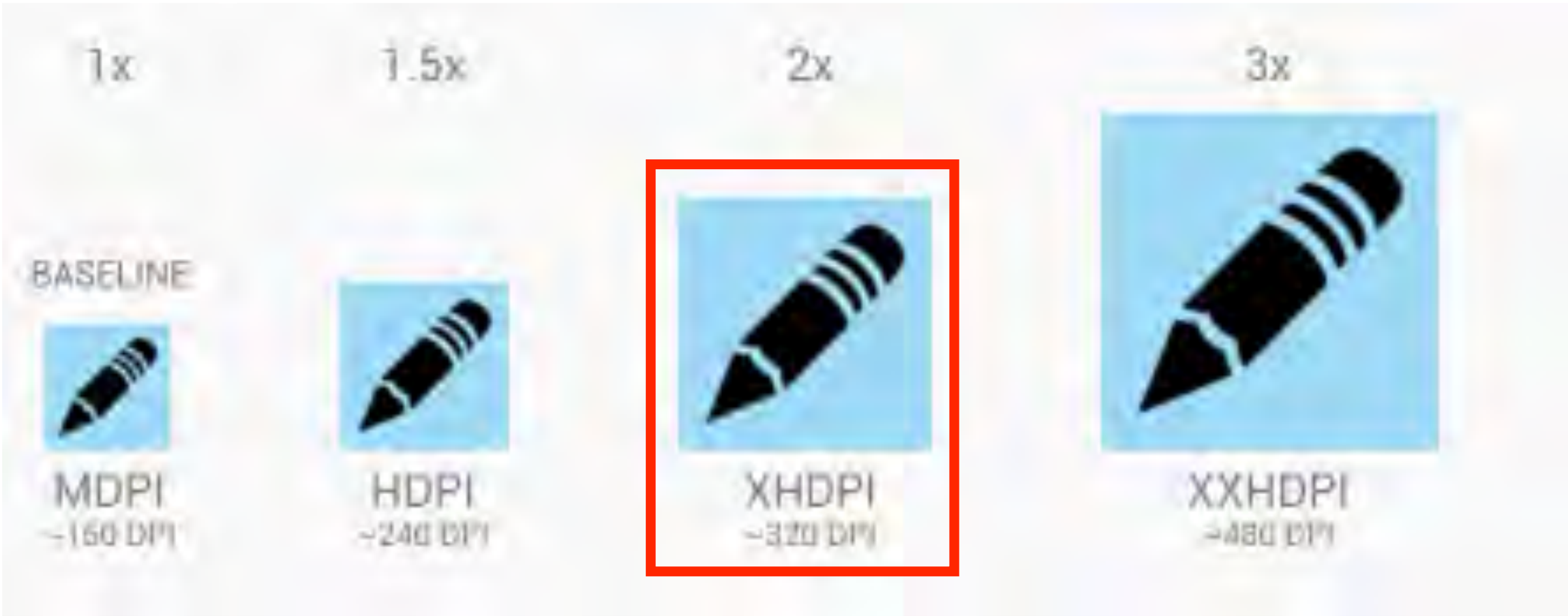
Optimize your application's UI by designing alternative layouts for some of the different size buckets, and provide alternative bitmap images for different density buckets.

- LDPI - low density (120 dpi)
- MDPI - medium density (160 dpi)
- HDPI - medium density (240 dpi)
- XHDPI - medium density (320 dpi)
- XXHDPI - medium density (480 dpi)

Different DPIs for different display categories

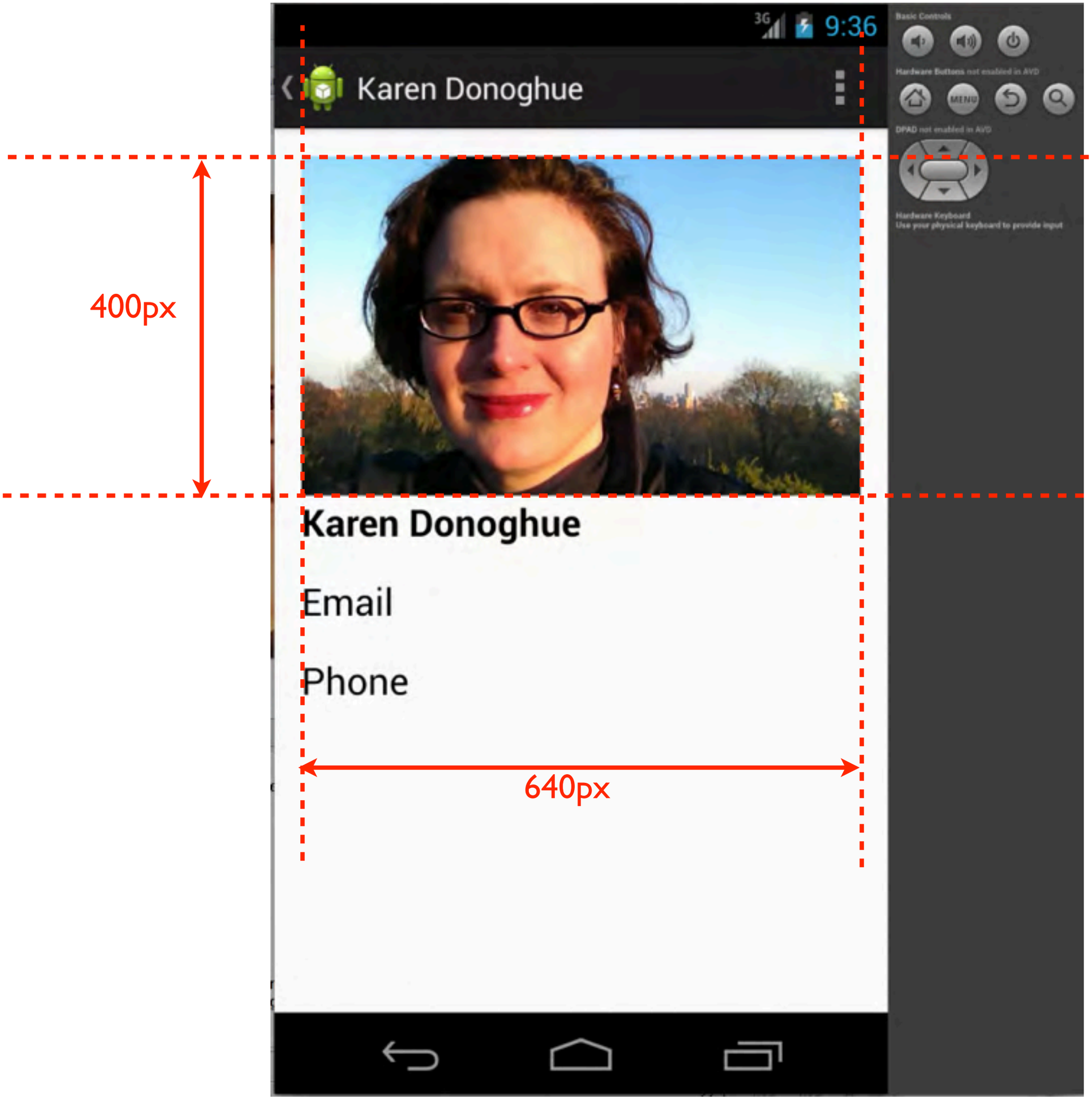


Different DPIs for different display categories



Dummy Data: Assignment2_People.zip

Name	Date Modified	Size	Kind
ContactDetailsPhotos	Today 6:51 PM	--	Folder
xhdpi (640px x 400px)	Today 4:36 PM	--	Folder
karen_boston.png	Aug 25, 2013 5:06 PM	450 KB	Portable Network Graphics image
karen_train.png	Aug 25, 2013 5:04 PM	498 KB	Portable Network Graphics image
max1.png	Aug 25, 2013 5:04 PM	437 KB	Portable Network Graphics image
max2.png	Aug 25, 2013 5:15 PM	343 KB	Portable Network Graphics image
max3.png	Aug 25, 2013 5:14 PM	265 KB	Portable Network Graphics image
ContactImageThumbnails	Today 6:50 PM	--	Folder
xhdpi (96px x 96px)	Today 3:29 PM	--	Folder
karen_train.png	Today 3:08 PM	96 KB	Portable Network Graphics image
karen.png	Today 3:07 PM	95 KB	Portable Network Graphics image
max1.png	Today 3:05 PM	86 KB	Portable Network Graphics image
max2.png	Today 3:09 PM	90 KB	Portable Network Graphics image
max3.png	Today 3:10 PM	92 KB	Portable Network Graphics image
Names	Today 2:55 PM	--	Folder
Names	Today 2:55 PM	937 bytes	Rich Text Document


9:36
Karen Donoghue
Basic Controls
Hardware Buttons not enabled in AVD
MENU
DPAD not enabled in AVD
Hardware Keyboard
Use your physical keyboard to provide input
400px
Karen Donoghue
Email
Phone
640px

Android vs iOS

- Android supports many DPIs and iOS supports two (Normal and Retina)
- Remember that both Android and iOS will scale available image assets if you are missing the correct DPI resources but they might appear blurry
- Question: What does XDPI represent in Android display densities?

Add your own images

- Feel free to add your own people photos
- This app is a "mockup" (class project for non-commercial use) so you can also use a service like http://uifaces.com
- This allows you to create a grid of faces to use for your app if you prefer
  - NOTE: You will need an image editor to slice and resize the photos

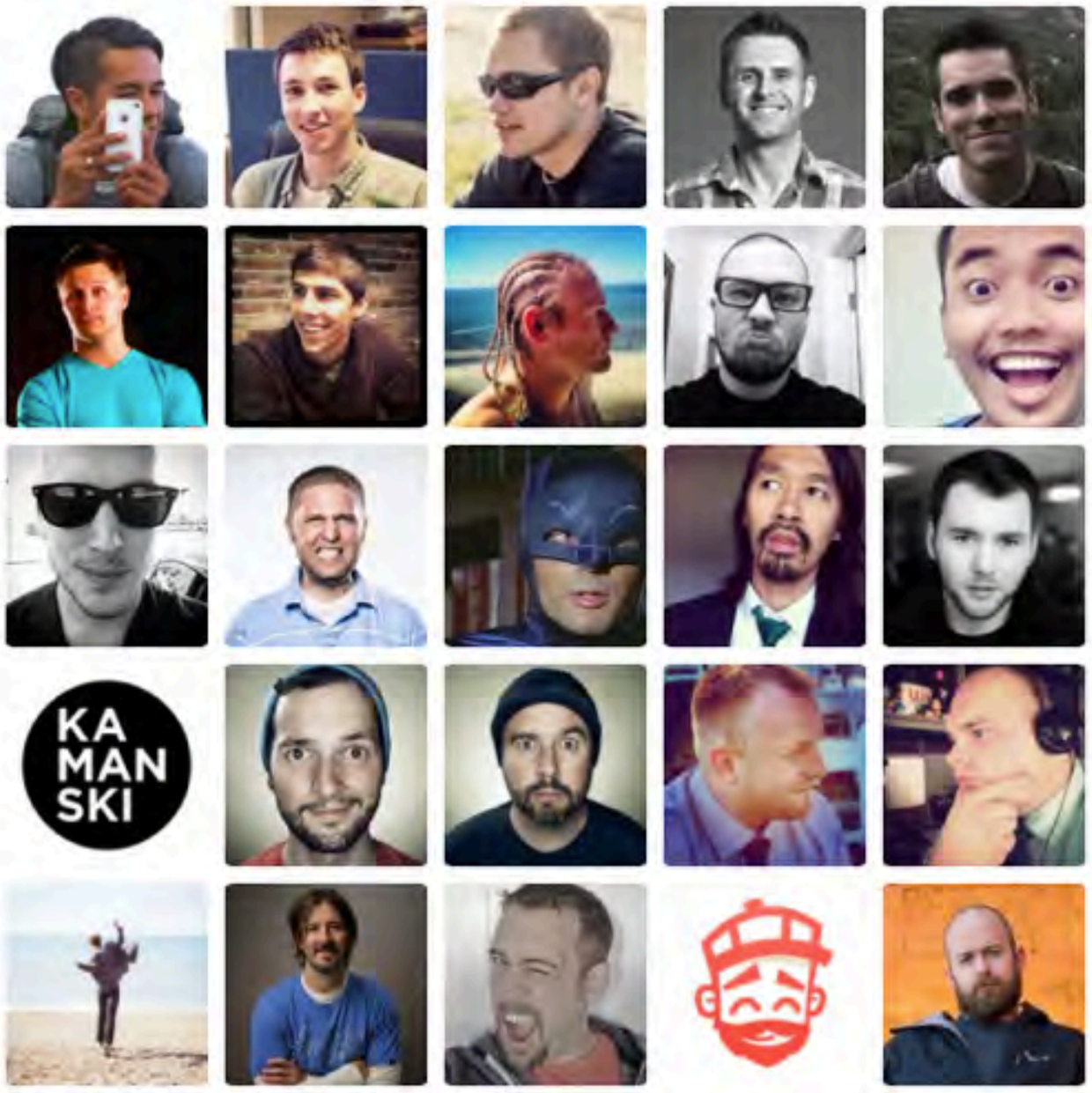
User Inter Faces

uifaces.com

Reader

uiFaces

Tweet 2,620



Avatar Size: **96px**

Spacing: **4px**

Border Radii: **3px**

What is this about?

Finding sample avatars for interface design can be super annoying. To use, adjust the avatar grid however you like, and take a screenshot. If you're on a mac, hit ⌘ + ⇧ + 4 to select the screenshot area from the guides.

Want your face here? Let me know

Android

- Our class will use: Studio - http://developer.android.com/sdk/installing/studio.html

- To see other dev tools you can check out the SDK + ADT plug in:
  - http://developer.android.com/tools/index.html

Activity and UI

- http://www.i-programmer.info/programming/android/5914-android-adventures-activity-and-ui.html
- Activity is the code that works with a UI screen defined by the View
- The Activity is the Java code that does something and the View provides the user interface (UI)

Android app consists of

- Activity
  - the Java code that does something
    - an Activity is the code that works with a UI screen defined by the View
- View
  - provides the UI

Intent

Interacting with Other Apps

An Android app typically has several activities. Each activity displays a user interface that allows the user to perform a specific task (such as view a map or take a photo). To take the user from one activity to another, your app must use an `Intent` to define your app's "intent" to do something. When you pass an `Intent` to the system with a method such as `startActivity()`, the system uses the `Intent` to identify and start the appropriate app component. Using intents even allows your app to start an activity that is contained in a separate app.

An `Intent` can be *explicit* in order to start a specific component (a specific `Activity` instance) or *implicit* in order to start any component that can handle the intended action (such as "capture a photo").

Android Tutorials/Resources

- Android Adventures - Getting Started With Android Studio
  - http://www.i-programmer.info/programming/android/5887-android-adventures-getting-started-with-android-studio.html
- For Assignment 2 please **read, view and learn** these sections
  - Getting Started With Android Studio
  - The Activity And The UI
  - Building The UI and a Calculator App
  - Lifecycle and State
  - Basic Controls And Events

Also please read/view

- Videos and tutorials in the Class Syllabus under Android and Android Studio

Android Studio demo

Android Studio

Android Studio makes creating Android apps a lot easier (limitations: OnClick event)

- An app has at least one Activity and this defines a screen layout and a behavior.
- The screen layout is controlled by XML markup file, Main_Activity.xml (usually) stored in the res directory.
- Drag-and-drop designer allows UI creation without having to work directly with XML.
- App behavior controlled by a Java file, MainActivity.java (usually) stored in the java directory. You can edit the code in the Java file directly in Android Studio.
- Run apps on emulator based AVD or Android device connected to laptop
- You need to create at least one AVD and start it running before trying to test your app

Sample media/assets

- Get the ZIP file Assignment2_People.zip from Karen or Aaron

Please bring your app to class

- Be ready to demo it in class