Android UI design patterns

Android UI design patterns

- Building applications that provide the right functionality and have good performance is one aspect of good application design
  - However, building a good UI that enables a user to get their tasks done quickly is just as important
  - In this lecture we will look at some of the design advice that is offered by google in relation to UI design. Much of the slide you see here are heavily based on the following documentation
    - https://developer.android.com/design/patterns/index.html

Gestures supported by android

- Android supports a number of core gestures by default.
  - Taking advantage of gestures that the user already knows will reduce the learning curve of your application
  - Will also reduce the amount of controls that you need to display on screen
  - There are eight gestures that are supported by default in android

Gestures supported by android

- Touch: single press and lift which triggers the default action for the given item.
- Long Press: enters data selection mode, gives the user the opportunity to select multiple items.
  - Will generally have an associated context bar. User presses then holds until they recieve vibration feedback before releasing
- Swipe or drag: touch followed by a movement then a lift. Used with overflowing content.
  - A swipe is quick and has momentum, while a drag is slower and more precise

Gestures supported by android

- Pinch open: for zooming in. Two fingers touch and then spread apart
- Pinch close; for zooming out. Two fingers touch and then move closer together
- Long press drag: used for rearranging items in a container. User holds down on an item and then has the option of moving it before releasing
- Double touch: two touches in quick succession for scaling up and down by a standard amount

Gestures supported by android

- And finally the double touch drag
  - Similar to the double touch but permits user scaling
  - Iniated by a single touch followed in quick succession by a drag
  - Up drag increases scale while down drags reduce scale

Application structure

- An android application will be structured into three main forms of views.
  - Top level views: the different views that your app supports, either shows different views of the same data or different pieces of functionality
  - Category views: give your user a chance to go deeper into the data they have stored in an application
  - Detail/edit views: where data is created and modified

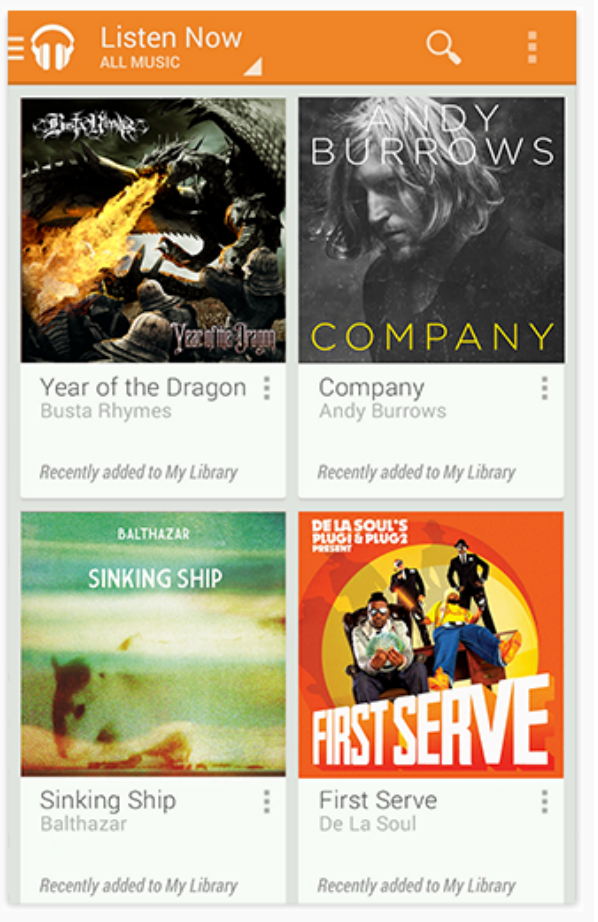
Different application types

- There are many different forms that an application can take in android here is a list of the three most common types
  - Apps built around a single activity (e.g camera or calculator)
  - Apps that switch between many different activities without deeper naviagation (e.g. the phone app)
  - Apps that have a broad set of data views and also deep navigation (e.g. gmail)

Top level views

- Views that should be considered carefully as it will be one of the first views seen when a user starts your application
  - Most applications now will show content on their top level views as a means of letting the user interact with content immediately
  - e.g. see Google play music app on next slide
  - The idea is to be visually engaging and to have a user interact with their content quickly i.e. a minimum of required effort

Top level views



Switching between top level views

- Generally to switch between top level views it is a good idea to use something like fixed tabs or spinners
  - Fixed tabs are always present on screen and generally you expect your users to flick between views frequently
  - Spinners are generally used if you do not wish to give up screen space to a set of tabs
    - Or if your user is switching to different views of the same set of data (e.g. a calander)

Category views

- Generally used in applications containing lots of data
  - Categories are used to organise and partition data, making it easy for users to find the data they need
  - Tabs can be used here as well particularly tabs that scroll from side to side but restrict yourself to 5-7 tabs at most
    - Examples of such categories include the play store app and also the contacts app

Detail views

- Where all data is created and modified in an application
  - Time should be taken to consider what the most common actions to be performed in a detail view are
  - The ones that are considered most common should be presented to the user first when the detail view is opened
  - Other options should be easily accessible either by scrolling or hidden away in a menu

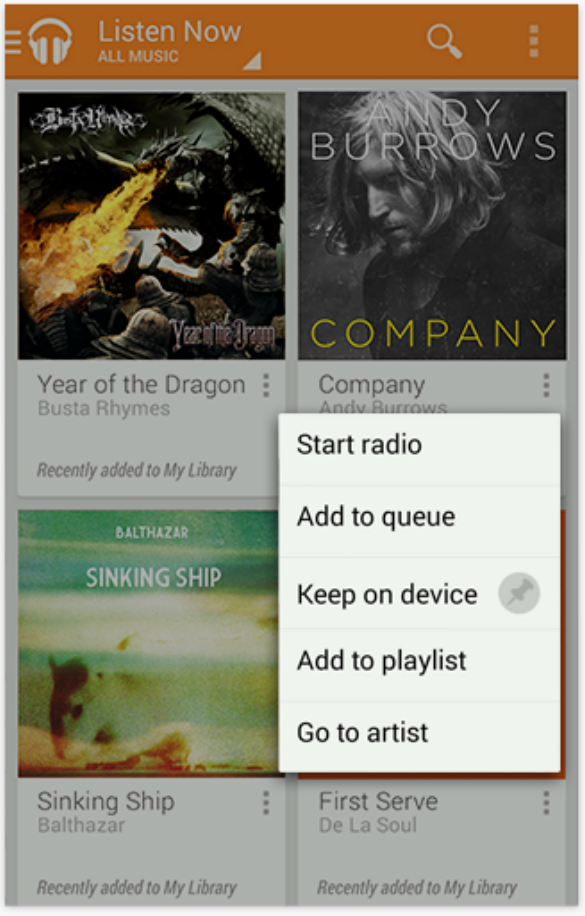
Navigation between detail views

- A special point must be mentioned here in that users may wish to switch between different items in the same detail view
  - e.g. swiping through emails in the gmail app
  - Permitting the user to scroll through multiple items without having to go back to a category view and then back into a detail view
  - If the list happens to be large like in a magazine or book then a thumbnail view may also be provided to aid navigation (e.g. zinio and kindle)

Cutting through hierarchies

- Sometimes it also helps to provide access to common actions on a top level view to prevent users from having to wade through hierarchies
  - An example would be google play music where there is a menu item attached to each individual piece of content
  - Permitting the user to do common actions without having to navigate down
  - Reducing the amount of effort on the user

Cutting through hierarchies



Navigation with up and back

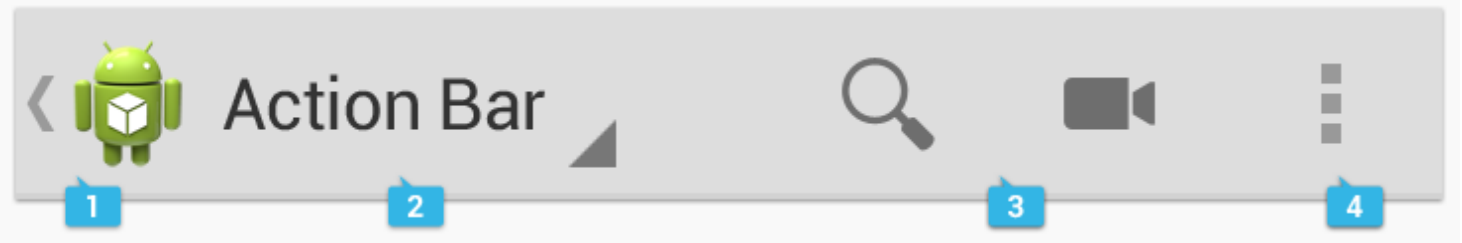
- Android generally provides two methods by which to navigate back through an application
  - There is the back button (either hardware of software. For navigating through activities in reverse chronological order
  - And a up button (software usually implemented as part of a UI in the top left of the action bar) which redirects to the previous activity in the hierarchy
    - The differnce between the two is that the up button will keep a user in your application where as the back button may switch between different applications

Action bar

- Dedicated UI control at the top of your application that is generally there for the lifetime of the app
  - Generally the most important actions are placed on the action bar.
  - Also used to enable consistent navigation and switching between views within the same application
  - Will provide a way for you to hide excess actions that are rarely accessed

Organisation of the action bar

- The action bar will look similar to that you see below and has the following components
  - 1) App icon, a visual indication of what app the user is in, also contains the up button (seen on the left) for navigating back a step in the hierarchy (not present on root activity)
  - 2) view control, users switch between different application views here, can be replaced by a tab view under the action bar if that is preferred



Organisation of the action bar

- The action bar will look similar to that you see below and has the following components
  - 3) action buttons, show the most important actions (most used) here. Those that do not fit will go into the overflow on the right (the three vertical dots)
  - 4) action overflow, any excess and less used actions are stored here

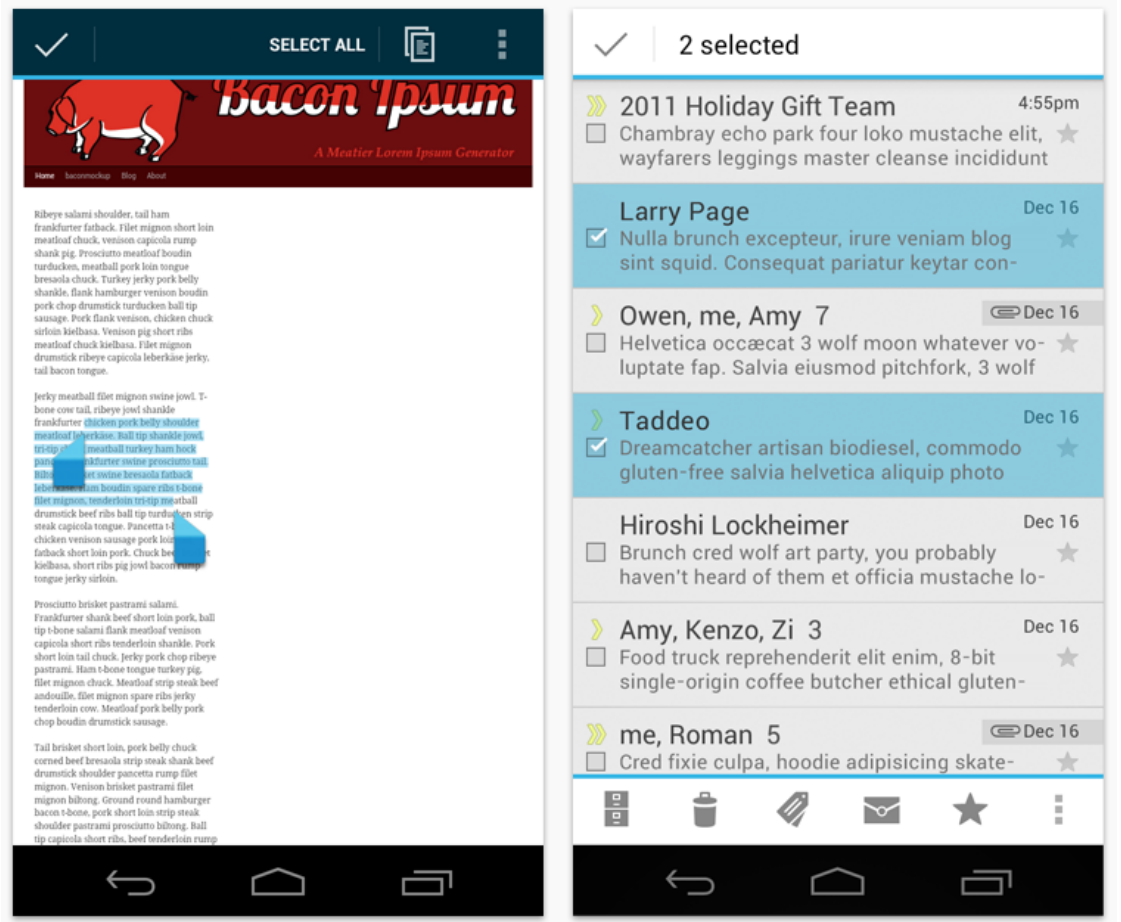
The FIT scheme

- The FIT scheme (Frequent, Important, Typical) is used to determine what actions should be prioritised and displayed
  - Frequent asks if the action is used at least 7/10 visits to this activity or if it is used several times in a row
  - Important asks if the action is something that is a selling point of your application
  - Typical asks if the action is considered important in similar applications

Contextual action bars

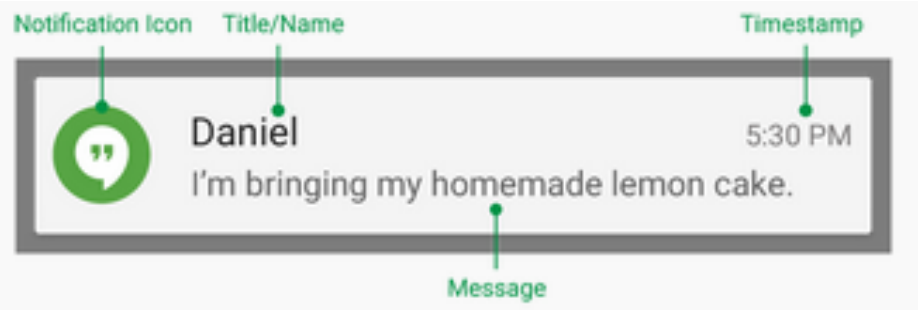
- A contextual action bar is a temporary action bar that is used to overlay the original action bar
  - Generally used when a user is in an action that involves selecting data or text
  - e.g. text selection in a browser or multiple message selection in gmail
  - The action bar will disappear when the user is finished performing that action

Contextual action bars



Notifications

- Notifications are used to inform users of important events in your application
  - Like new messages, calendar events, birthdays etc
  - Notifications can be a powerful way to keep users informed at all times
  - However your application must be careful to choose what notifications it will send and also how the notifications are structured



Notification anatomy

- A notification consists of 4 core components
  - The first is a notification icon that will give the user a quick visual indication as to what the notification is about. Also displayed in the notification bar at the top left
  - The title gives a quick indication of what the notification is about
  - The time stamp details when the notifcation occured
  - And the message gives more detail about the notification itself

Expanded notification layout

- It is also possible to show an expanded form of the notifications that will provide more detail to the user
  - This could be showing the first few lines of a message
  - Or a picture or screenshot that was taken
  - Gives the user an option to expand or contract the notification layout there and then and decide if it can be dismissed or acted upon now

Notification actions

- It is also possible to attach actions (small buttons) to each notification
  - Examples would include a reply action to an email or a snooze button on an alarm
  - Enables the user to respond to an action quickly
  - Can prevent the need to enter the application to dismiss or mute events

Heads up notifications

- Should only be used for high priority notifications that must have an immediate or near immediate response
  - e.g. accepting or dismissing an incoming call
  - These will be displayed to the user immediately if they are currently using a device
  - Regardless if the application has hidden all access to the notification bar on screen

Navigation with notifications

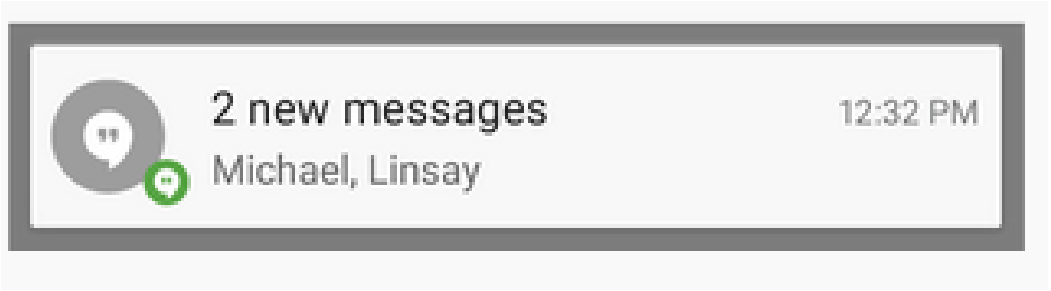
- When the user touches the notification body without touching an action your application should navigate them to the appropriate place to act on that notification
  - For example in gmail hitting a message notification will bring the user to the full content of that email.
  - Special attention must be paid if you have collapsed multiple notifications into one.
    - In this case you will need to provide a summary of all notifcations and permit the user to choose between them.

Notification priorities

- There are 5 notification priorities in android
  - Max, high, default, low and min
  - Max should be used for critical urgent notifications that needs a quick resolution
  - High is generally used for important communications
  - Default is for those that don't fall into any of the other four categories
  - While low and min will show up at the bottom of the list, the user wishes to be notified about these things but they are not important

Summarise your notifications

- Try not to clutter up the notification bar with multiple instances of the same notification type
  - We do this by summarising them
  - e.g. in the example below there are two separate hangout messages however they have been collapsed into one has the user has not responded to either yet



Notification style in lollipop onwards

- As material design is now used in android 5.0 onwards it is a good idea to design your notification icons to reflect this design
  - Generally your notification icons in the notification bar should consist of pieces that are fully white and others that are transparent
  - Shape is used to distinguish between the different notification types
  - Here is a few examples of this notification style