Mobile Application Programming: Android

Data Driven UI
Activities

- Apps are composed of activities
- Activities are self-contained tasks made up of one screen-full of information
- Activities start one another and are destroyed commonly
- Apps can use activities belonging to another app
Creating a Custom Control

- Create subclass of View class
- Override:
  - `onDraw(Canvas c)`
  - `onMeasure(int wMeasure, int hMeasure)`
- Add listener interface and listener property for the interesting events the control generates and call on... methods when events occur
ListView

- Lists data provided by an Adapter
- Use ArrayAdapter or a custom class to provide data
- Set `OnItemClickListener` to react to clicks on rows
GridView

- Lists data provided by an Adapter
- Use ArrayAdapter or a custom class to provide data
- Set OnItemClickListener to react to clicks on rows
- Call setNumColumns to change the grid resolution
Providing Data

- Data-driven views require an Adapter
- Adapter is an interface only
  - Requires a concrete class
  - Can (and should) create your own implementations
- Is the secret to potentially-infinite data-set sizes
- Retrieves data as associated views come on-screen
- This requires the Adapter to be somewhat complex
ArrayAdapter

- Concrete implementation of Adapter
- Add arbitrary objects to have them displayed in text-based view rows by way of the toString method
- Must specify a resource ID for a row layout on creation
  - Create your own, or specify android.R.layout.simple_list_item_1
- Requires that data be provided up-front, rather than the more flexible generic Adapter
Adapter Activity

- Activity class itself can implement the Adaptor interface
- Implement each method returning 0 or false except:
  - getCount - return number of objects
  - getItem - return item that represents row
  - getViewTypeCount - return 1 (or more)
  - getView - return a view to represent the data item
- Reuse passed view or create a new view
- Fill in the view with the data from getItem