Creating the UI

- Three approaches
  - in XML
    - using UI builder of XML editor
  - in code
    - calls to factory methods to create and add components
  - combined
    - define fixed UI in XML
    - modify/add/delete components for customization at execution

Components
- layouts
- widgets

Interaction
- handle events

Layouts
- Control the placement of other components
- Aka: Container, ViewGroup
  - LinearLayout
    - elements laid out in column (vertical) or row (horizontal)
  - RelativeLayout
    - elements laid out relative to each other and the parent container
  - AbsoluteLayout (deprecated)
    - elements laid out according to absolute positioning (x,y)
  - FrameLayout
    - elements are stacked on top of the previous
  - TableLayout
    - elements laid out by (row,column)
  - GridLayout
    - elements laid out by (row,column) with spanning and overlay
Widgets

- Objects for user interaction
  - Aka: View, Control
  - **TextView**
    - read-only text box
  - **EditText**
    - editable text box
  - **ListView**
    - selectable scrolling list of other views
  - **Spinner**
    - selectable list of text items
  - **Button**
    - a clickable button
  - **CheckBox**
    - bi-stable checkable button
  - **RadioButton**
    - set of mutually exclusive checkable buttons

Event Handling

- User interaction leads to an event
  - e.g., clicking on a button
- Callbacks to handle event
  - method that is called when event occurs
  - callback method is registered with originator of the event
- Event handler
  - an anonymous inner class
    - create ad hoc as handler
  - the Activity itself
    - implement interface and register method of class
    - specify method in XML
  - a separate class
    - implement interface, create and register in Activity

Welcome Message App

- UI
  - XML
    - attributes
      - width/height (wrap vs match)
    - id
    - text
  - UI Builder
Event Handler
- implements Button.OnClickListener
- method `onClick` handles event
  - register via `setOnClickListener`
- accessing UI components
  - `R` class
    - automatically generated
    - `findViewById`
  - Context method
    - `getText`, `setText`
    - `View` parameter identifies the actual component generating the event

Toast
- Transient message to user
  - doesn’t require user to dismiss
  - information messages
    - app doesn’t depend on user reacting to message
- `Toast` factory method `makeText`
  - `Context`, `CharSequence`
  - `message`
  - duration
- `show` method
Starting Activities

- An activity can start another activity
- Message passing via Intent
  - explicit
    - specify class of activity to be started
    - only good for activities within same app
  - implicit
    - specify action required
    - Android runtime finds class
    - may include data
    - Extras
- Example - welcome message handled by separate Activity
  - layout files
  - manifest
  - classes

EditText

- Editable text box
- Attributes
  - layout_width, layout_height
  - hint
  - textSize
  - autoText
  - password
  - height/width
  - inputType

EditText Events

- Can handle EditText events
- Welcome app with reactive EditText
  - UI
    - only EditText & TextView
  - event handling
    - onKeyListener interface
    - onKey method
      - ACTION_UP - key up
      - KEYCODE_ENTER - enter key
      - result
        - true - event handled
**CheckBox**

- Bi-stable checkable selector
- Methods
  - isChecked(), setChecked(boolean), toggle()
- Example Order Food
  - select then press calculate button
  - interrogate checkboxes state
- Event Handling
  - onCheckedChanged (CompoundButton c, boolean checked)
    - occurs when checkbox state changes
    - can determine which checkbox and after change state

**RadioButton**

- Grouped into RadioGroup
  - mutually exclusive checkboxes
- Methods
  - isChecked(), toggle(), check(), getCheckedRadioButtonId()
- Event Handling
  - onClick – when button clicked (selected)
  - onCheckedChanged – when button state changes
- Example Hotel Selector
  - all buttons in group use same handler
  - when click can determine which was selected

**Layouts**

- Control layout of widgets
- Are laid out within screen or another layout
- Layout attributes
  - specified for child (widget)
  - vary by layout manager
  - layout...
  - fine tune layout of children within container
  - setLayoutParams(…)

---

---

---

---
LinearLayout

- Widgets laid out in rows or columns
- Attributes
  - orientation
    - vertical (rows), horizontal (columns)
    - also setOrientation(...)  
  - layout_width, layout_height
    - wrap_content - layout resizes to accommodate children
    - match_parent - layout fills parent
  - padding
    - space between boundary and content
    - also setPadding(...)  
Widget Attributes

- layout_width, layout_height
  - width/height of widget in container
  - match_parent
    - size of container
  - wrap_content
    - sizes to fit contents
  - explicit (px, dp, sp, pts, in, mm)
  - also setLayoutParams(...)  
- layout_weight (0.0..1.0)
  - proportional amount of extra space applied to widget
    - proportion of sum of weights
    - also setLayoutParams(...)  
- gravity
  - layout of content within widget
    - center_vertical, center_horizontal, center, left, right, top, bottom...
    - multiple values separated by | (e.g. top|right)
    - also setGravity(...)  
- layout_gravity
  - layout of widget within container
    - as above
    - also setLayoutParams(...)  