Views Presentation

Goals

- Learn how to build a complete layout with nested LinearLayouts
- Know about some common Views for input and display
- Classes and methods introduced in this chapter
    - LinearLayout, RelativeLayout, EditText, CheckBox, ImageView, Spinner

Layouts

- The layout element is responsible for laying out its child elements
    - Determines width and height of each child element, and how to position them in relation to one another. Different layout classes uses different attributes and rules when performing this operation.
- Layout classes are derived from the common base class ViewGroup
- Layout rules are specified as attributes on child elements with prefix "layout_"

Layouts in this course

- RelativeLayout
  - Default layout in the latest project template
  - Child elements are layed out relative to the layout itself, or other child elements
  - Not very good for proportional layouts
- LinearLayout
  - Easy to use, fewer rules and attributes to learn
  - Very good at handling percentages for laying out children proportionally
  - We'll try and limit ourselves to LinearLayout in the course for simplicity reasons

LinearLayout attributes

- android:orientation
  - Specify "horizontal" or "vertical" depending on in which direction all children should be layed out
- Nesting
  - Create LinearLayout inside of a LinearLayout
  - Forms a "Grid"
  - Usually the orientation changes
  - The child LinearLayout has child attributes like other child Views
  - Not recommended for performance reasons if it can be avoided (with other layouts)

LinearLayout layout attributes for children

- android:layout_width
  - wrap_content to let the contents of the view decide the width
  - match_parent to have the width of the child extended to the maximum width of the layout
- android:layout_height
  - Same as layout_width, but for the height of course. Usually for a layout with horizontal orientation it doesn't make any sense to have any child specify layout_width as fill_parent and vice versa.
- android:layout_gravity
  - If the layout_width/layout_height hasn't extended the size of the child view to the full width/height of the layout, this value will determine to which edge to align the child view.
- android:layout_margin
  - Specify a value here to have extra space between the child view's contents and any surrounding child views. Is also available in parts such as layout_marginLeft, layout_marginTop etc.

Automatic expansion with layout_weight

- One or more views can be automatically distributed evenly across the width or height
  - Can also be used to make a view fill the remaining space of a container
  - For some reason you don't specify percentages for layout_width/layout_height, but instead use layout_weight
- Usage
  - Set the layout_width/layout_height to "0px"
  - Set the layout_weight to a number
- The value of layout_weight only have meaning in comparison with other views in the same container that also have a layout_weight specified

EditText

- Input field where user can enter text, numbers etc
- android:inputType
  - Defines what type of content can be typed in the field
  - Also means that different virtual keyboards can be displayed
  - Many options exist, like "text", "textMultiLine", "textPassword", "number", "phone"
  - Can also be combined with flags like "textAutoCorrect"
- Read/write current value in Java

```
String value = _textView.getText().toString();
_textView.setText("Hello, world!");
```

CheckBox

- Input field with a checkmark and text
- android:checked
  - The default value of the field, "true" och "false"
- CheckedTextView
  - Can be used instead of CheckBox if you want the check mark to the right of the text
- Read/write current value in Java

```
if (_checkBox.isChecked())
  _checkBox.setChecked(false);
```

ImageView

- Display an image
- android:src
  - The image to display. Unfortunately http/web address are not possible here. Must be a Drawable.
- android:contentDescription
  - Textual description/replacement of the image. Important to specify for accessibility reasons (people with visual disabilities).
- android:scaleType
  - How to display the image with regards to the size of the view. One example is "fitXY" where the image is stretched to fit the view both horizontally and vertically. Default is "center" where the image size is not changed and it is centered in the view.

Spinner

- Drop list of options
  - Default from Android 4.0 is to show drop list in the same view, for older versions the list of options may be shown in a dialog popup.
- android:entries
  - An array xml resource identifier. The xml resource file will contain all strings in the list.
- /res/values/array.xml

```
<resources>
  <string-array name="countries">
    <item>Sweden</item>
```

- android:prompt
  - Title of dialog popup when such is shown for choosing one of the options
- android:spinnerMode
  - Can be used to force popup dialog on newer versions using value "dialog".

View

- Use just the base View object when a drawable background is all you need
- Can also be useful when padding/extending empty areas in a user interface
- Or to simluate lines/dividers in the user interface

More information

- Book references
  - The Busy Coder's Guide 3.7
    - Using XML-based layouts, page 69
    - Working with containers, page 93
    - Using selection widgets, page 133