CMSC 436 Lab 9

Developing Complex User Interfaces

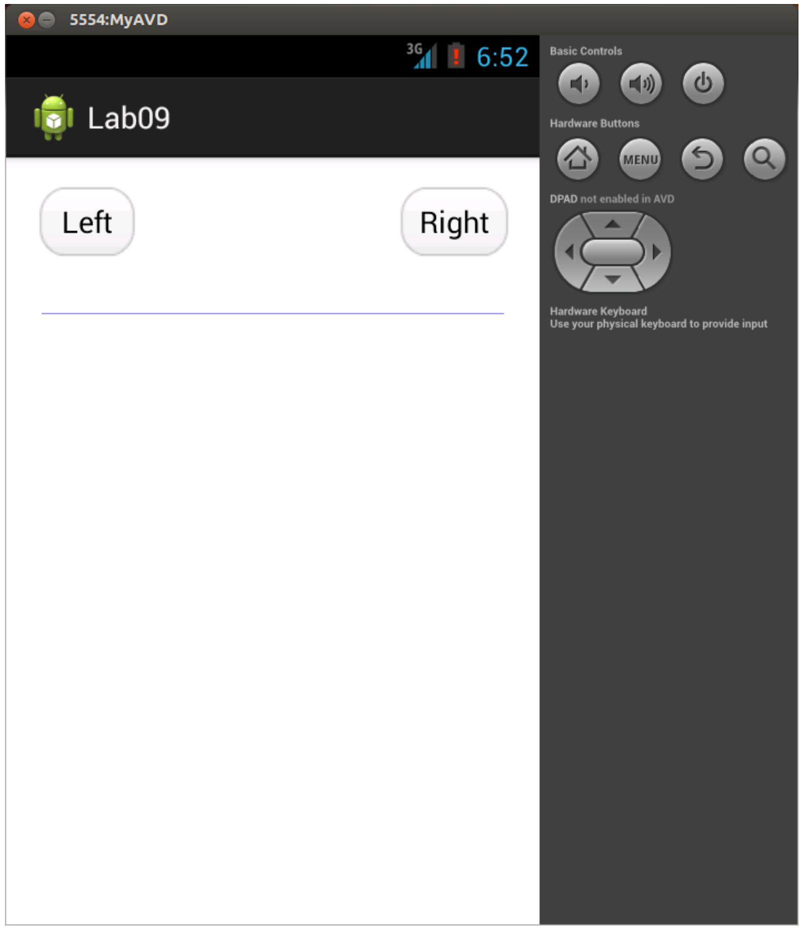
Overview

- For this lab you will practice using different Android user interface components, including
  - Button (with custom background)
  - RelativeLayout
  - Floating context menu
  - A custom component
- The information needed to do this lab can be found on the Android developer site at

  http://developer.android.com/guide/topics/ui/

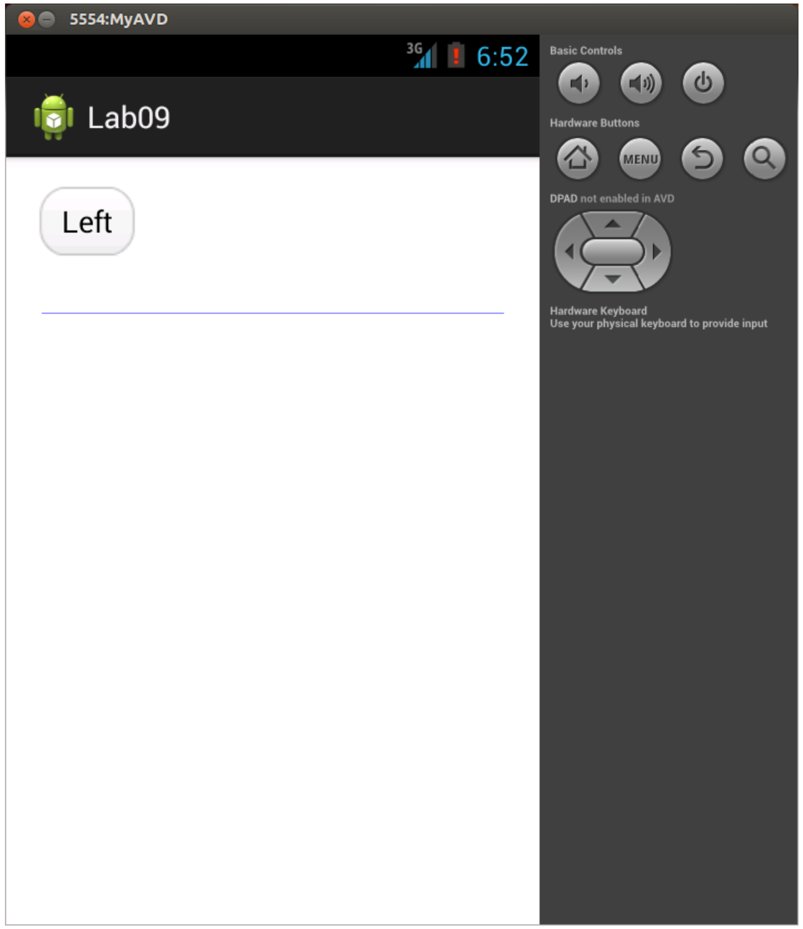
Interface

- Your main activity should contain “Left” and “Right” buttons, positioned as shown, above a custom component LinedEditText



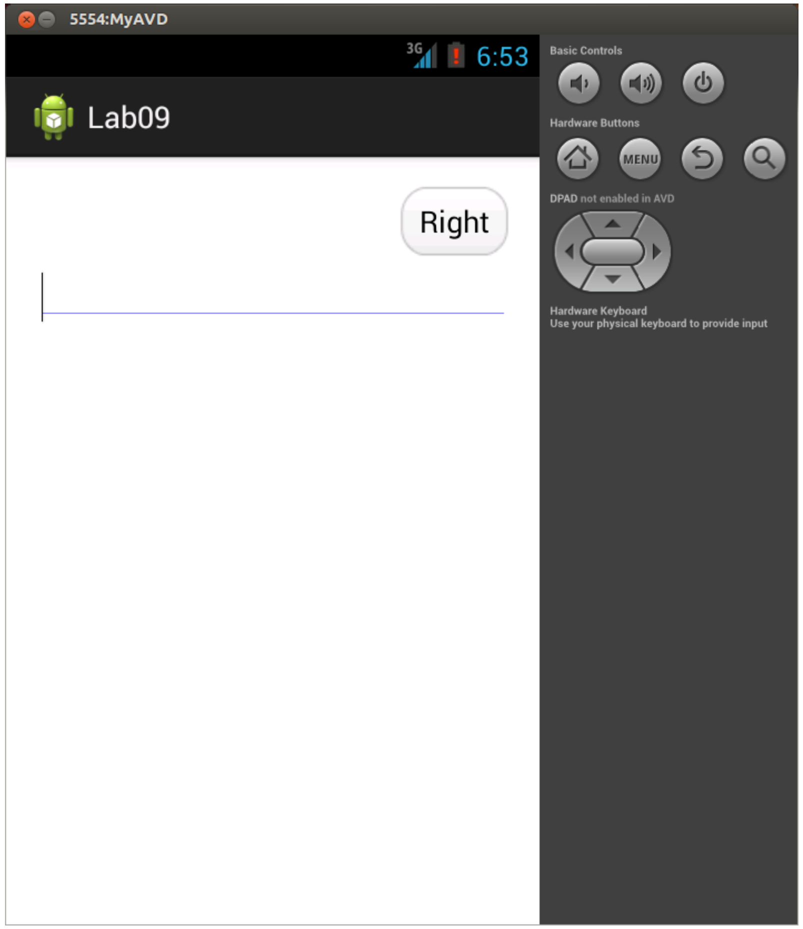
Interface

- Clicking the “Left” button should toggle the visibility of the “Right” button between visible / invisible



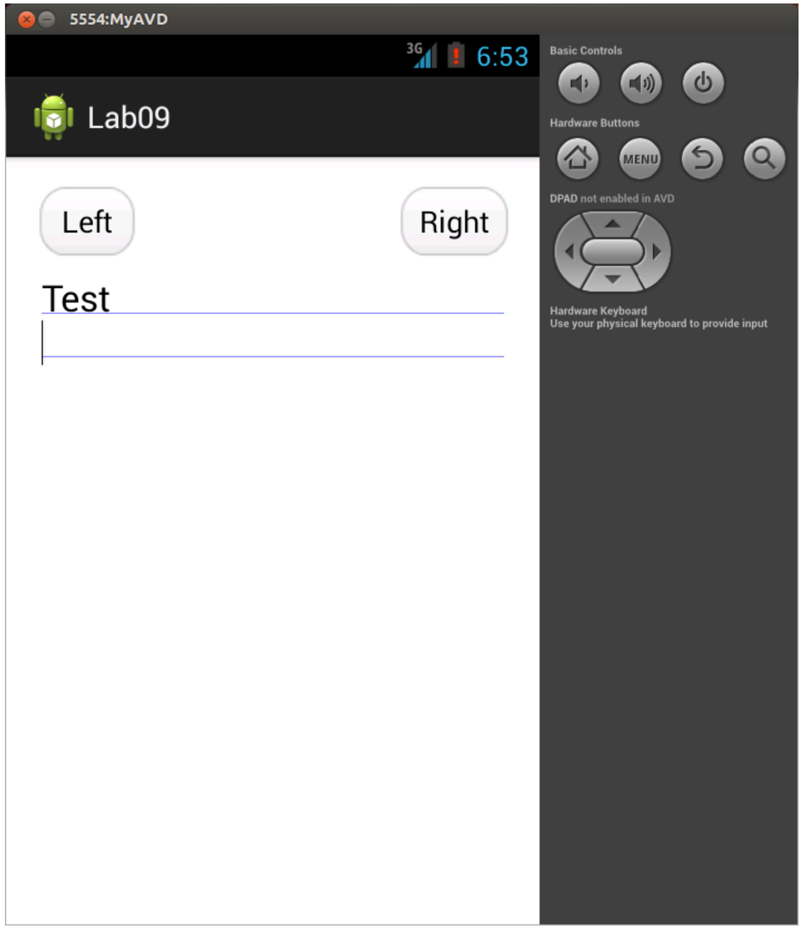
Interface

- Clicking the “Right” button should do the same, but to the “Left” button



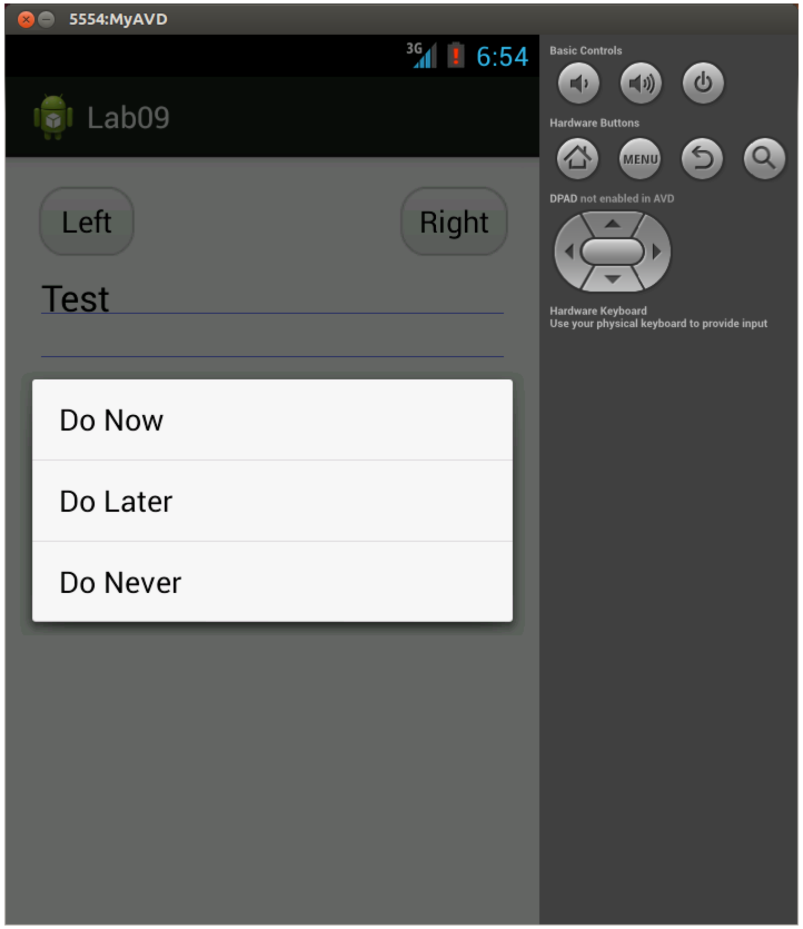
Interface

- The user should be able to type lines of text into the LinedEditText
- Each line of text should have a blue underline



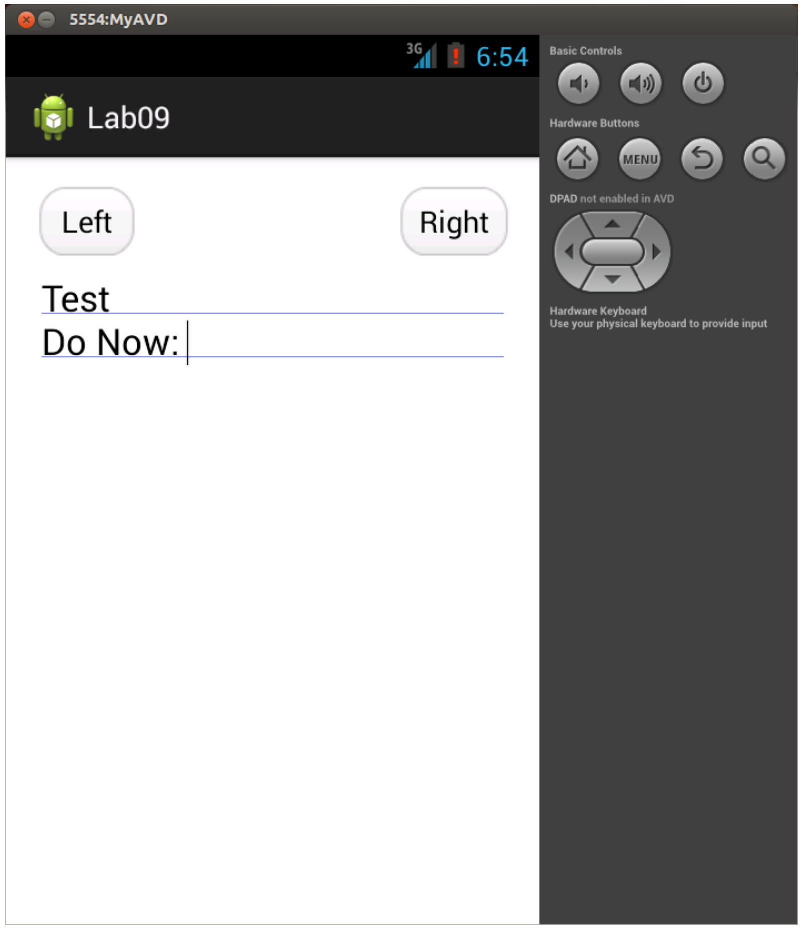
Interface

- Doing a long-click on the LinedEditText should bring up a floating context menu with items as shown



Interface

- Selecting one of the items should append a string composed of the name of the item plus a semicolon to the end of the existing text



Test
Do Now:

RelativeLayout

- Up to this point most of your labs have used only LinearLayouts. LinearLayouts can be nested to position Views in a variety of arrangements, but nesting layouts too deeply can adversely impact performance. An alternative to nested LinearLayouts is a RelativeLayout, which can often achieve the same effect using only a single layout.
- For this lab you should use only a single RelativeLayout and no other layouts.

Buttons with Custom Background

- Android Buttons support using custom images for the background
- A different image is used depending on whether the Button is in the pressed, focused, or default state
- The images must be in the 9-patch format, which is a standard png with a 1 pixel border around it that indicates what parts of the image should be stretched when the Button is scaled

Buttons with Custom Background

- Android provides a tool to make images of this type, but but they can also be found on the web, for example

  http://android.svn.wordpress.org/branches/reader/res/drawable/

  - grey_wp_button_down.9.png
  - grey_wp_button_active.9.png
  - grey_wp_button.9.png

Buttons with Custom Backgrounds

- For this lab you should make both of the buttons use custom backgrounds for all three states
- You can make your own 9-patch images or download some from the web
- To see how to implement custom backgrounds, look at the Input Controls / Buttons subsection of the User Interface section on the Android developer site

Custom Component

- Custom components implementing arbitrary functionality can be created by extending the View class, but you can often reuse functionality of existing components by extending them instead
- For this lab you will use the LinedEditText component that extends EditText
- You will also further modify its functionality

Custom Component

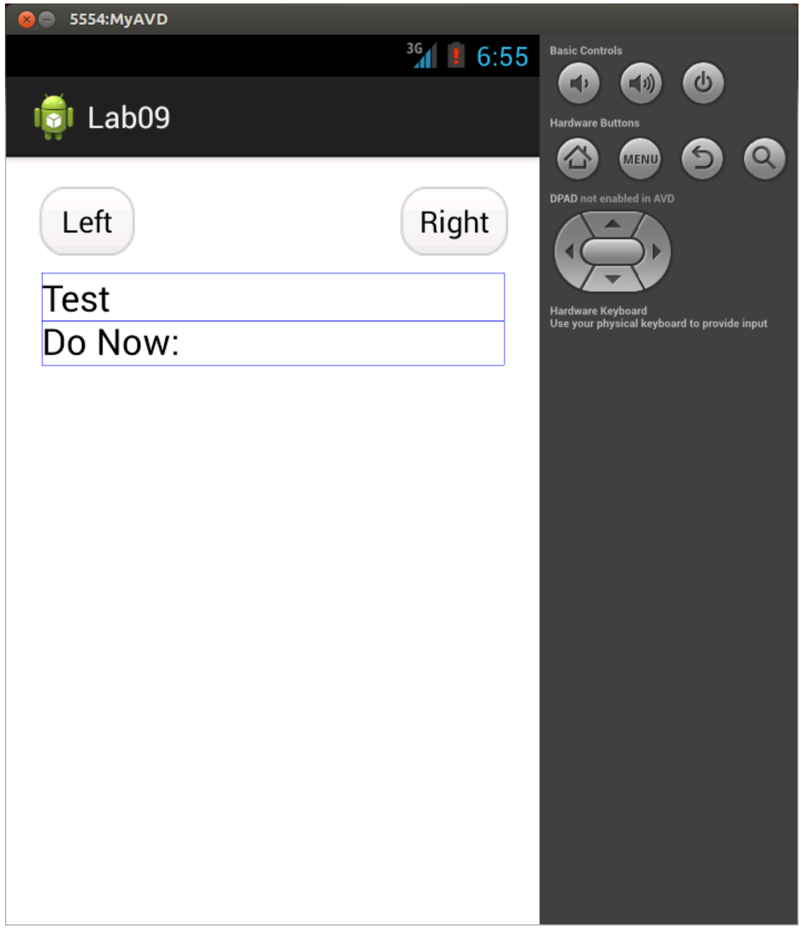
- To get the LinedEditText component, first download the Android SDK samples
- Open the Android SDK Manager and make sure that “Samples for SDK” is installed
- Then go to the NotePad example in the samples directory of the Android SDK and look at NoteEditor.java
- The NoteEditor class contains an inner class called LinedEditText- copy this inner class into your main activity class

Custom Component

- The NotePad example also include the layout file note_editor.xml which shows how you can add this custom component to your layout xml
- The LinedEditText component works by overriding the onDraw method of EditText to draw a blue line under each line of text
- Look through this method to see how it works, then modify it to instead draw a blue box around each line of text

Custom Component

- The final result should look like this



Test
Do Now:

Floating Context Menu

- To see how to implement floating context menus, look at the Menus subsection of the previously linked User Interface section on the Android developer site
- You should first define the menu in xml, and then add the appropriate callbacks to your main activity