SketchPad - Part 1

In this lab you will be creating an app which allows you to draw on a 32x32 RGB LED board through the Arduino. Your app will have two different user interfaces to choose from. The first one will be a simple interface where you will type in values to control the board. The second interface will be more complicated but will allow you control the board by drawing on a grid in the app. This week you will be implementing the simple interface.

Objectives:

- Learn how to use additional hardware with the Arduino
- Create own format for serial messages
- Create different user interfaces to accomplish the same task
- Show real world applications of embedded systems

Before You Start:

Download sketch_pad_shell_arduino.zip and open sketch_pad.ino with the Arduino IDE.

Download SketchPadShell.zip and import the project in Android Studio.

Connect LED board to Arduino if it is not already.

Arduino:

For the Arduino portion of this lab, you will be creating your own serial message format and writing code to interpret this format. Later you will be writing Android code to send a message in this format. While implementing the serial message think about how you sent data in the Morse Code Lab.

Your serial message should include the following information.

- A signal to erase the entire board.
- A signal to display a pre-loaded bitmap image.
- An x, y location of a pixel to modify.
- RGB values for the color of the pixel.

Erasing the Board:

Your message should contain a signal to erase the entire LED board. When this signal is activated, call the *blankEasel()* function to clear the board.

Displaying Pre-Loaded Bitmap:

This is very similar to how you erase the LED board. This time, when your signal is activated, call the *loadBitmap()* function.

Reading X, Y Location:

When you read the x and y values from your message, you should set the global variables cursorX and cursorY to those values. This is important because the drawing function relies on these values.

Reading RGB Values:

When you read the RGB values from your message, set the global red, green, and blue variables to their corresponding values. Again for use in the drawing function *(Note that the LEDs on the board only support RGB values from 0-7. This means that your message should only include RGB values that are also in this range).*

Drawing a Pixel:

This should be done last and only if you neither erased the LED board or loaded the bitmap. After you have set cursorX, cursorY, red, green, and blue, you can call drawDot() to set the pixel at (cursorX, cursorY) to the current color.

Android - Part 1 (Coordinate Editor):

The Android portion of this lab is divided into two parts. In each part, you will be creating a different user interface to draw on the LED board. In this part, you will be creating a simple interface which allows you to manually enter coordinates and RGB values to draw on the board.

Starting the Coordinate Editor:

When the SketchPad app is started, the user is presented with a screen with two buttons to choose which interface they want to use, Coordinate Editor or Sketchpad. In MainActivity.java, create a method to launch the CoordinateEditor activity. *(Hint: Remember that Intents can be used to start a new activity).*

Erasing the Board:

In CoordinateEditor.java, there is a method called *erase()* which is called when the “Erase” button is pressed. Complete this method, so that it will send the erase signal that you created earlier to the Arduino board. This should be done similarly to the *blinkLED()* method in the Morse Code Lab.

Displaying Pre-Loaded Bitmap:

Repeat the process you used to complete *erase()* method, this time to complete *loadBitMap()*.

Drawing a Pixel:

In this step, you will complete the *changeColor()* method. This method will get the x, y coordinates and RGB values from their corresponding EditTexts and send them to the Arduino to be displayed on the LED board. *(Remember that RGB values for the LED board must range from 0-7).*

Demo:

After you have completed these steps, upload the app to your Android device and connect it to your Arduino board.

Demo this part to your TA.