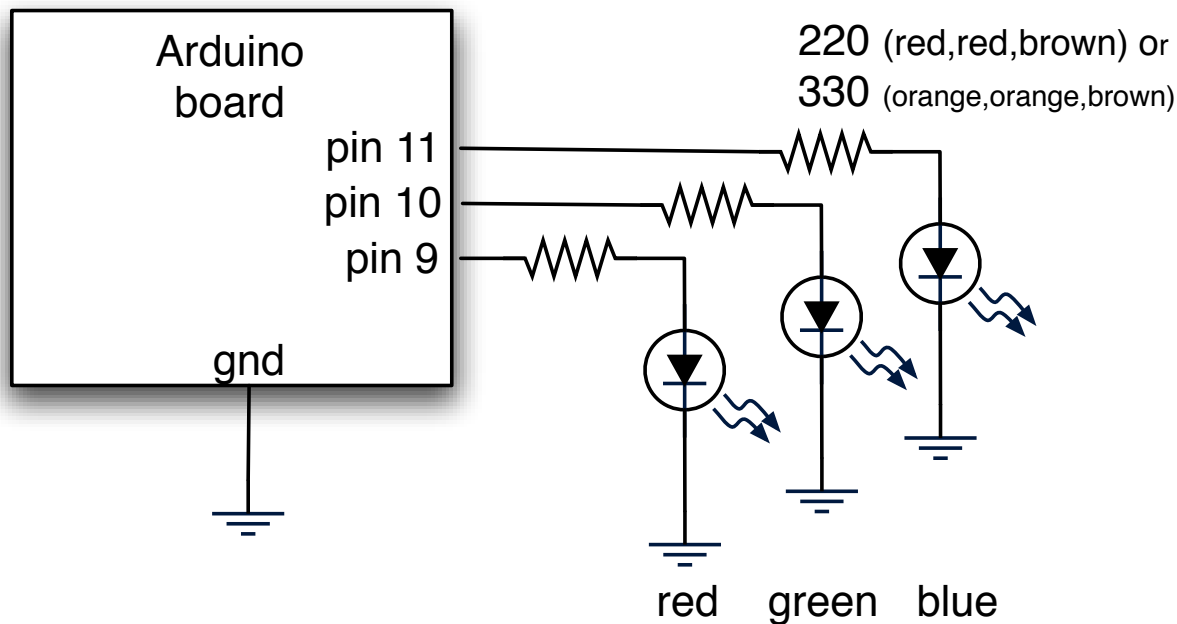


R,G,B LEDs

Three PWM outputs and *three* primary colors.
Just screams to be made, doesn't it?

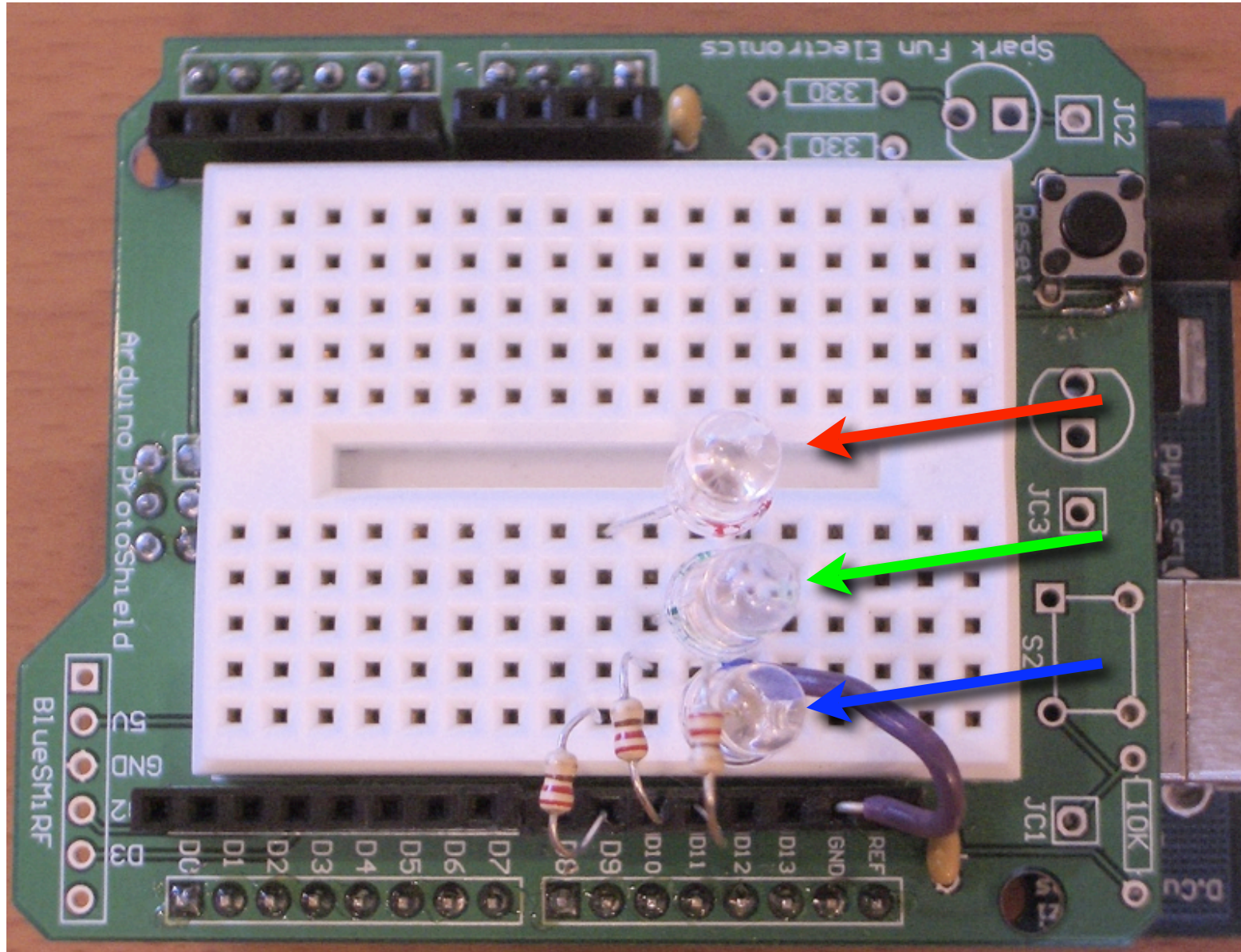


With RGB you can
make any color
(except black)

Put back on the ProtoShield for this.

Use either the 220 or 330 ohm resistors in your kit, if you don't have enough of one or the other I have lots more 220 if you need them

R,G,B LEDs



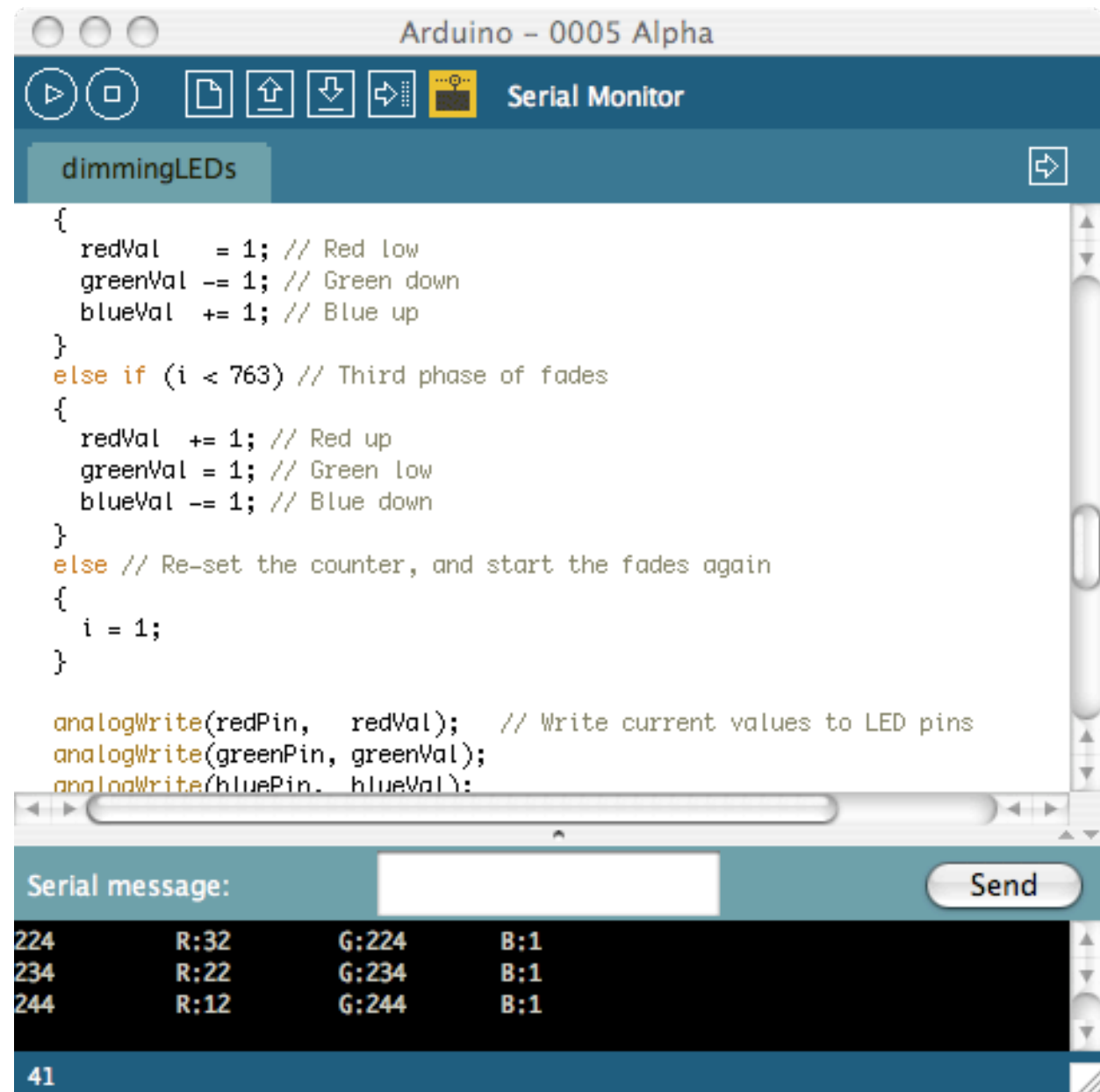
Cut leads of resistors and LEDs to make for a more compact circuit.
Also, less likely to short against itself.

RGB Color Fading

“dimmingLEDs”

Slow color fading
and mixing

Also outputs the
current color values
to the serial port



The screenshot shows the Arduino IDE Serial Monitor window titled "Arduino - 0005 Alpha". The window displays the code for the "dimmingLEDs" sketch. The code is as follows:

```
{
  redVal    = 1; // Red Low
  greenVal -= 1; // Green down
  blueVal  += 1; // Blue up
}
else if (i < 763) // Third phase of fades
{
  redVal  += 1; // Red up
  greenVal = 1; // Green low
  blueVal -= 1; // Blue down
}
else // Re-set the counter, and start the fades again
{
  i = 1;
}

analogWrite(redPin,  redVal); // Write current values to LED pins
analogWrite(greenPin, greenVal);
analogWrite(bluePin,  blueVal);
```

The Serial Monitor output shows the following data:

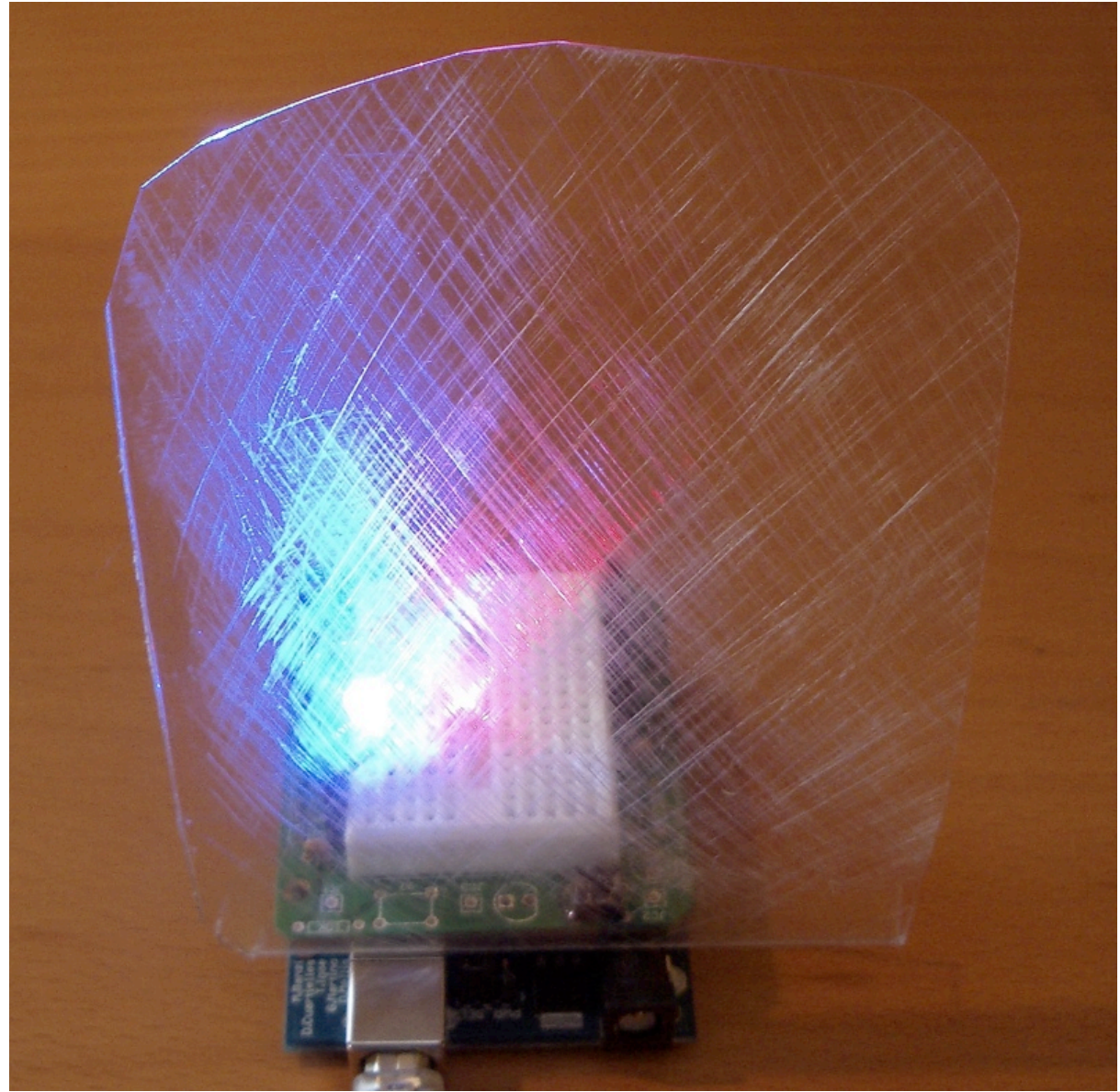
Serial message:	R	G	B
224	32	224	1
234	22	234	1
244	12	244	1

The Serial Monitor also shows a "Send" button and a line number "41" at the bottom.

This sketch is located in the handout.
It just ramps up and down the red, green, & blue color values and writes them with `analogWrite()`
from <http://www.arduino.cc/en/Tutorial/DimmingLEDs>

Mood Light

Diffuser made from
piece of plastic
scratched with
sandpaper



Also, can use plastic wrap scrunched up to make an interesting diffuser.

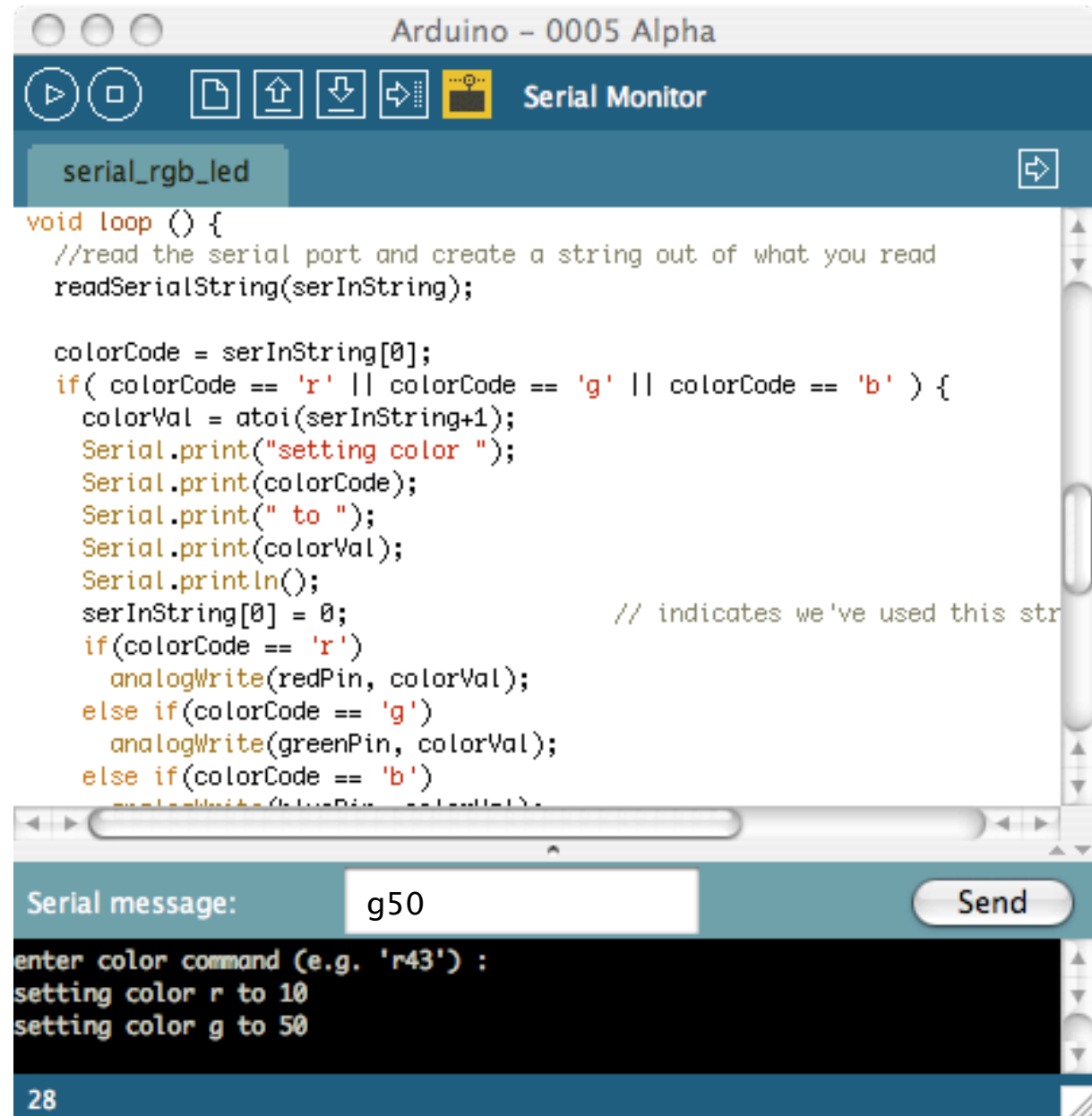
Serial-controlled RGB

"serial_rgb_led"

Send color
commands to
Arduino

e.g. "r200", "g50", "b0"

Sketch parses what
you type, changes
LEDs



```
Arduino - 0005 Alpha
Serial Monitor
serial_rgb_led

void loop () {
  //read the serial port and create a string out of what you read
  readSerialString(serInString);

  colorCode = serInString[0];
  if( colorCode == 'r' || colorCode == 'g' || colorCode == 'b' ) {
    colorVal = atoi(serInString+1);
    Serial.print("setting color ");
    Serial.print(colorCode);
    Serial.print(" to ");
    Serial.print(colorVal);
    Serial.println();
    serInString[0] = 0; // indicates we've used this str
    if(colorCode == 'r')
      analogWrite(redPin, colorVal);
    else if(colorCode == 'g')
      analogWrite(greenPin, colorVal);
    else if(colorCode == 'b')
      analogWrite(bluePin, colorVal);
  }
}
```

Serial message:

```
enter color command (e.g. 'r43') :
setting color r to 10
setting color g to 50
```

28

This sketch is located in the handout.

Color command is two parts: colorCode and colorValue

colorCode is a character, 'r', 'g', or 'b'.

colorValue is a number between 0-255.

Sketch shows rudimentary character string processing in Arduino

Reading Serial Strings

- New Serial function in last sketch:
“Serial.available()”
- Can use it to read all available serial data from computer
- Great for reading strings of characters
- The “readSerialString()” function at right takes a character string and sticks available serial data into it

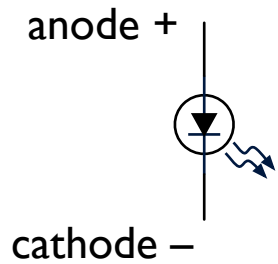
```
//read a string from the serial and store it in an array
//you must supply the array variable
void readSerialString (char *strArray) {
    int i = 0;
    if(!Serial.available()) {
        return;
    }
    while (Serial.available()) {
        strArray[i] = Serial.read();
        i++;
    }
}
```

Pay no attention to the pointer symbol (“*”)

Must be careful about calling readSerialString() too often or you'll read partial strings

RGB LEDs

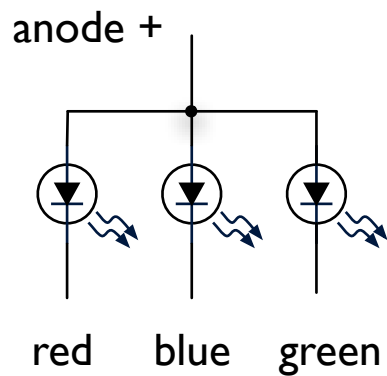
Normal LED



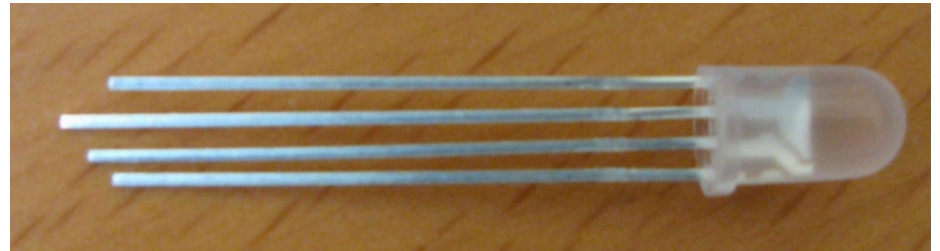
anode +
cathode -



RGB LED



red cathode -
anode +
blue cathode -
green cathode -



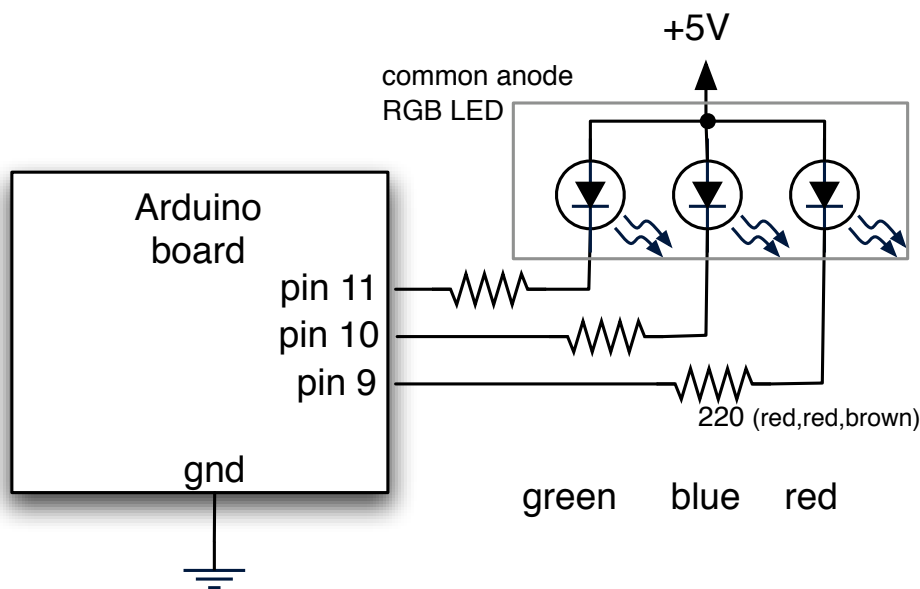
actually 3 LEDs in one package

RGB LED, aka "tri-color LED"

Common-anode RGB LEDs are much more available than common-cathode. This is why we're changing around the logic.

Color Mixing

With just 3 LEDs you can make any* color



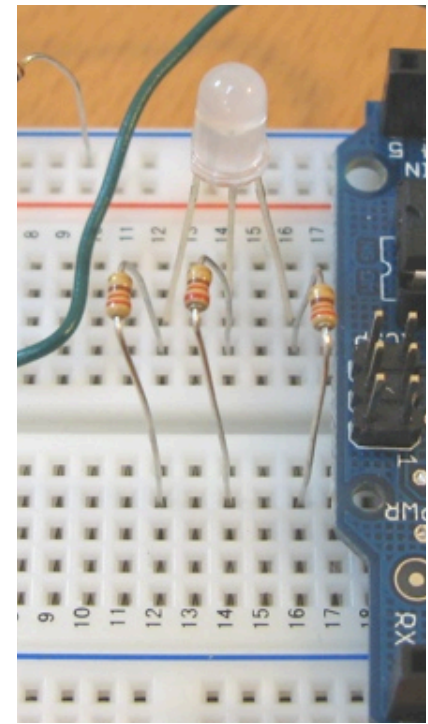
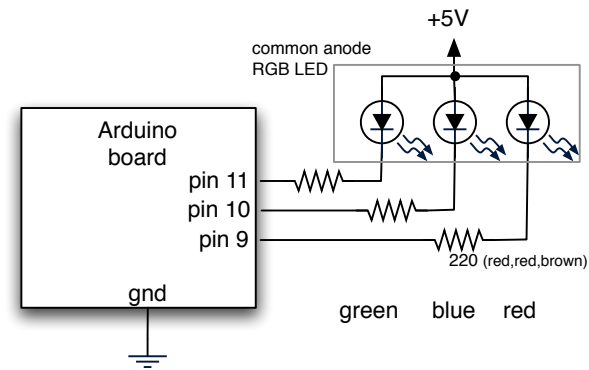
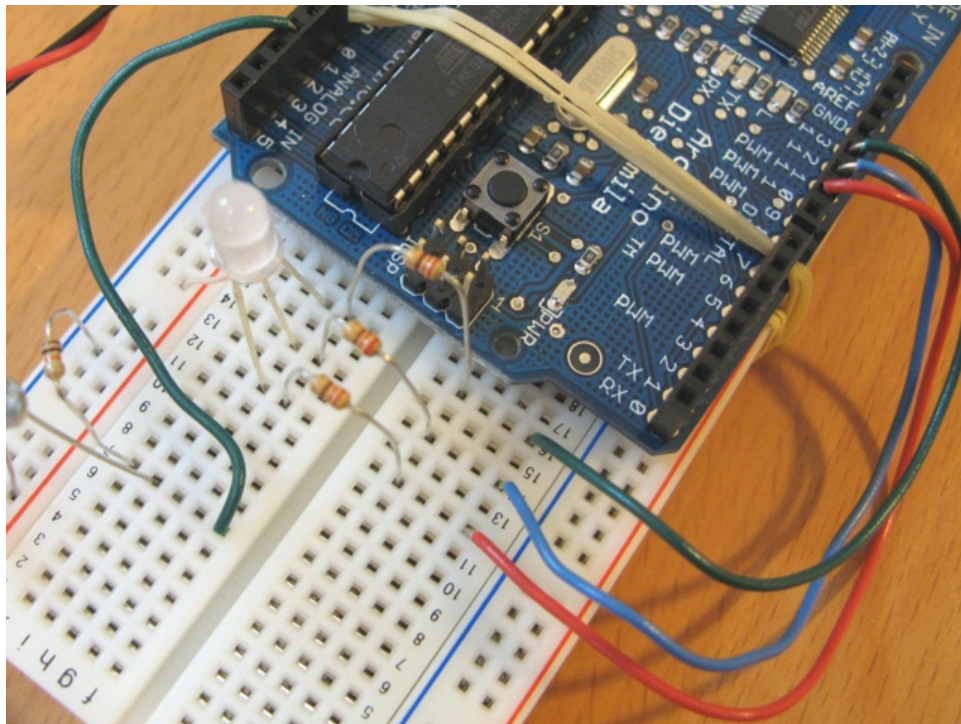
With RGB you can make any color (except black)

Mixing light is the additive color model

(paint is subtractive color, and can give you brown)

- *besides the additive/subtractive color difference, it's hard to get the mix to be just right for a variety of annoying reasons:
- the physics of LEDs mean that different color LEDs put out different amounts of light
 - our eyes respond non-linearly across the spectrum, i.e. we're more sensitive to green than red
 - the lenses in most RGB LEDs don't focus each color to the same spot

Laying out RGB LED Circuit



slightly bend the longest lead and plug it into the +5v (red) bus

plug remaining leads into rows (12,14,&16 here)

connect 220 (red-red-brown) resistors across middle to matching rows

run wires from resistors to pins 9,10,11 of Arduino, can color-code if you want

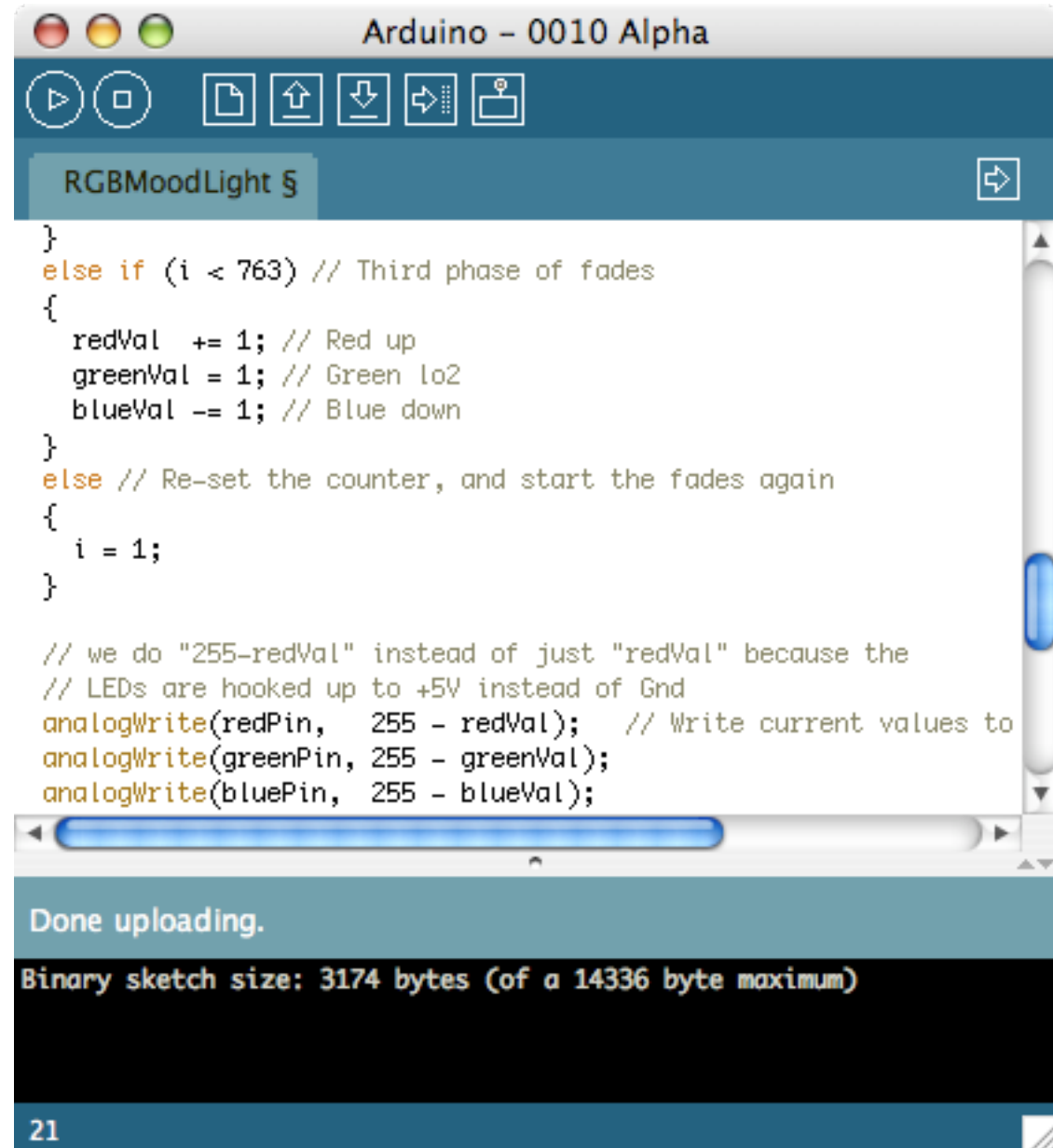
Ignore the green wire in the pictures, that's another circuit.
Keep the pot from last circuit if you can.

RGB Color Fading

“RGBMoodLight”

Slow color fading
and mixing

Also outputs the current
color values to the serial port



```
Arduino - 0010 Alpha
RGBMoodLight 5
}
else if (i < 763) // Third phase of fades
{
  redVal += 1; // Red up
  greenVal = 1; // Green lo2
  blueVal -= 1; // Blue down
}
else // Re-set the counter, and start the fades again
{
  i = 1;
}

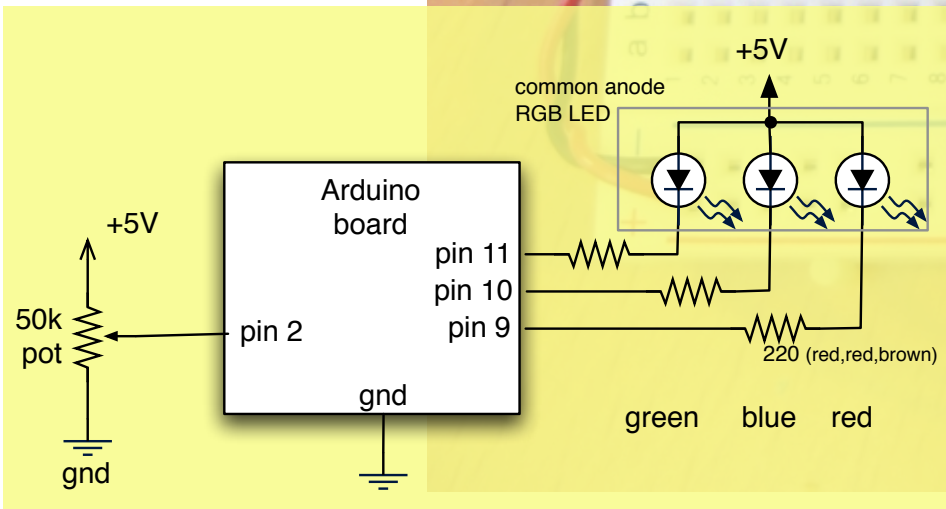
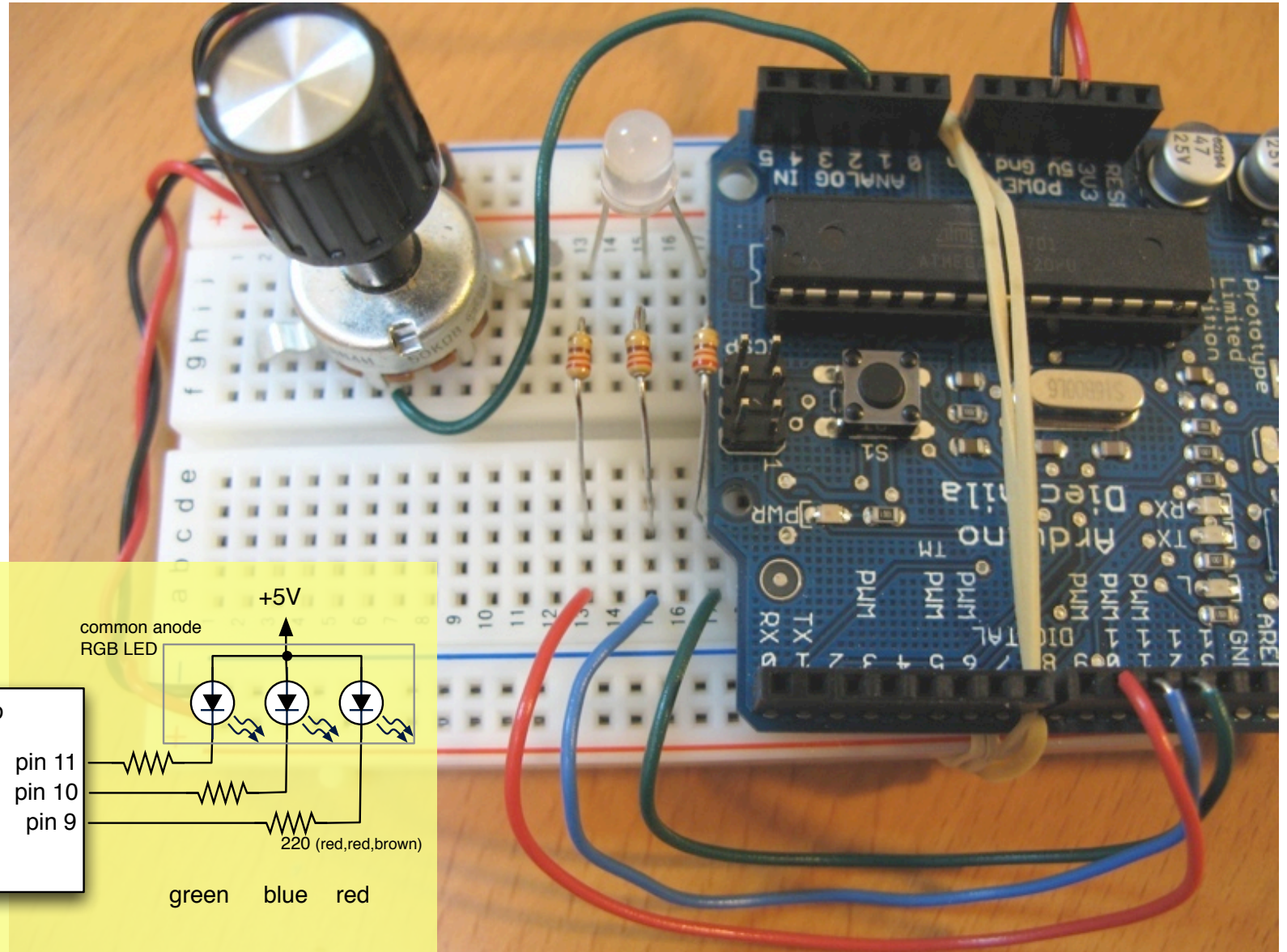
// we do "255-redVal" instead of just "redVal" because the
// LEDs are hooked up to +5V instead of Gnd
analogWrite(redPin, 255 - redVal); // Write current values to
analogWrite(greenPin, 255 - greenVal);
analogWrite(bluePin, 255 - blueVal);

Done uploading.
Binary sketch size: 3174 bytes (of a 14336 byte maximum)
21
```

This sketch is located in the handout.
We'll get to the serial port stuff in a minute.

It just ramps up and down the red, green, & blue color values and writes them with `analogWrite()`
from <http://www.arduino.cc/en/Tutorial/DimmingLEDs>

Pot-controlled RGB

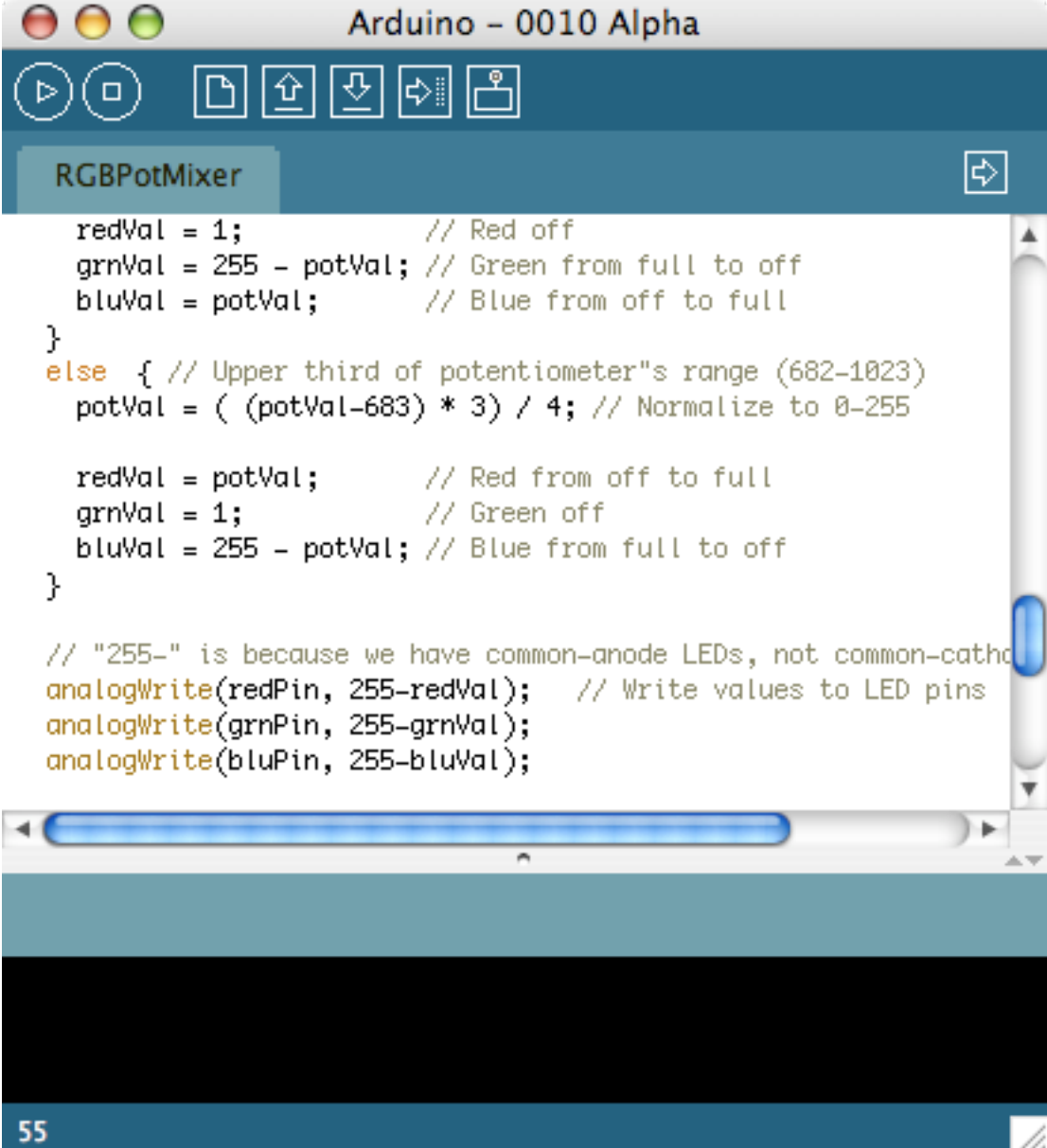


Pot-controlled RGB

“RGBPotMixer”

Use the pot from before to control the color mix

The code turns the single ranged input value into “sectors” where each sector is a color



```
Arduino - 0010 Alpha
RGBPotMixer
redVal = 1; // Red off
grnVal = 255 - potVal; // Green from full to off
bluVal = potVal; // Blue from off to full
}
else { // Upper third of potentiometer's range (682-1023)
  potVal = ( (potVal-683) * 3) / 4; // Normalize to 0-255

  redVal = potVal; // Red from off to full
  grnVal = 1; // Green off
  bluVal = 255 - potVal; // Blue from full to off
}

// "255-" is because we have common-anode LEDs, not common-cathode
analogWrite(redPin, 255-redVal); // Write values to LED pins
analogWrite(grnPin, 255-grnVal);
analogWrite(bluPin, 255-bluVal);
55
```

Also see “RGBPotMixer2” for a variation.
How would you change it to adjust brightness?