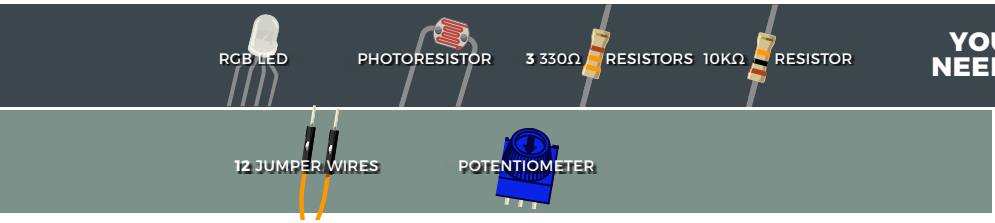
Circuit 1D: RGB Night-Light

In this circuit, you'll take the night-light concept to the next level by adding an RGB LED, which is three differently colored Light-Emitting Diodes (LEDs) built into one component. RGB stands for Red, Green and Blue, and these three colors can be combined to create any color of the rainbow!



NEW COMPONENTS

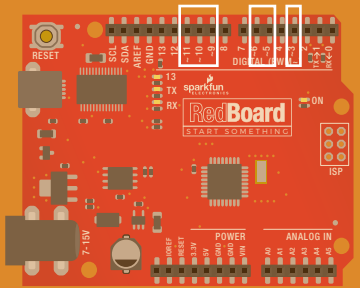
RGB LED: An RGB LED is actually three small LEDs — one red, one green and one blue — inside a normal LED housing. This RGB LED has all the internal LEDs share the same ground wire, so there are four legs in total. To turn on one color, ensure ground is connected, then power one of the legs just as you would a regular LED. Don't forget the current-limiting resistors. If you turn on more than one color at a time, you will see the colors start to blend together to form a new color.

NEW CONCEPTS

ANALOG OUTPUT (PULSE-WIDTH MODULATION): The `digitalWrite()` command can turn pins on (5V) or off (0V), but what if you want to output 2.5V? The `analogWrite()` command can output 2.5 volts by quickly switching a pin on and off so that it is only on 50 percent of the time (50% of 5V is 2.5V). By doing this, any voltage between 0 and 5V can be produced. This is what is known as Pulse-Width Modulation (PWM). It can create many different colors on the RGB LED.

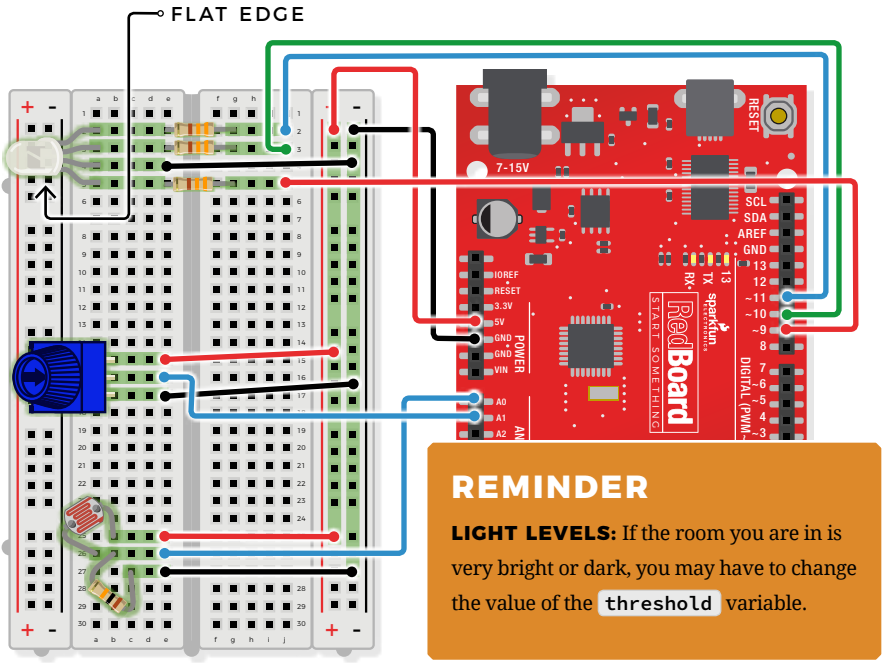
NEW IDEAS

PWM PINS: Only a few of the pins on the RedBoard have the circuitry needed to turn on and off fast enough for PWM. These are pins 3, 5, 6, 9, 10 and 11. Each PWM pin is marked with a ~ on the board. Remember, you can only use `analogWrite()` on these pins.



HOOKUP GUIDE

READY TO START HOOKING EVERYTHING UP? Check out the circuit diagram and hookup table below to see how everything is connected.



REMINDER

LIGHT LEVELS: If the room you are in is very bright or dark, you may have to change the value of the `threshold` variable.

CONNECTION TYPES ◆ REDBOARD CONNECTION ■ BREADBOARD CONNECTION

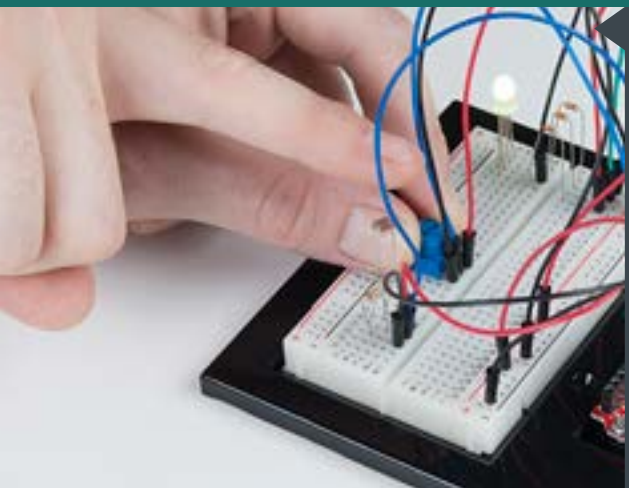
JUMPER WIRES	◆ 5V to ■ 5V(+) · ◆ GND to ■ GND (-) · ◆ D9 to ■ J5 · ◆ D10 to ■ J3 · ◆ D11 to ■ J2 · ◆ A0 to ■ E26 · ◆ A1 to ■ E16 · ■ E15 to ■ 5V(+) · ■ E17 to GND(-) · ■ E4 to ■ GND(-) · ■ E25 to ■ 5V(+) · ■ E27 to ■ GND (-)
RGB LED	■ A5(RED) + ■ A4(GND) + ■ A3(GREEN) + ■ A2(BLUE)
330Ω RESISTORS (ORANGE, ORANGE, BROWN)	■ E2 to ■ F2 · ■ E3 to ■ F3 · ■ E5 to ■ F5
10KΩ RESISTOR (BROWN, BLACK, ORANGE)	■ B26 to ■ C27
PHOTORESISTOR	■ A26 to ■ B25
POTENTIOMETER	■ B15 + ■ B16 + ■ B17

Open the Arduino IDE

Connect the RedBoard to a USB port on your computer.

Open the Sketch:
File > Examples > SIK_Guide_Code-V_4 > **SIK_CIRCUIT_1D-RGB NIGHT LIGHT**

Select **UPLOAD** to program the sketch on the RedBoard.



WHAT YOU SHOULD SEE

This sketch is not dissimilar from the last. It reads the value from the photoresistor, compares it to a threshold value, and turns the RGB LED on or off accordingly. This time, however, we've added a potentiometer back into the circuit. When you twist the trimpot, you should see the color of the RGB LED change based on the trimpot's value.

PROGRAM OVERVIEW

1	Store the light level from pin A0 in the variable **`photoresistor`**.
2	Store the potentiometer value from pin A1 in the variable **`potentiometer`**.
3	If the light level variable is above the **`threshold`**, call the function that turns the RGB LED off.
4	If the light level variable is below the **`threshold`**, call one of the color functions to turn the RGB LED on.
5	If **`potentiometer`** is between 0 and 150, turn the RGB LED on red.
6	If **`potentiometer`** is between 151 and 300, turn the RGB LED on orange.
7	If **`potentiometter`** is between 301 and 450, turn the RGB LED on yellow.
8	If **`potentiometer`** is between 451 and 600, turn the RGB LED on green.
9	If **`potentiometer`** is between 601 and 750, turn the RGB LED on cyan.
10	If **`potentiometer`** is between 751 and 900, turn the RGB LED on blue.
11	If **`potentiometer`** is greater than 900, turn the RGB LED on magenta.

CODE TO NOTE

ANALOG OUTPUT (PWM):

```
analogWrite(RedPin, 100);
```

The `analogWrite()` function outputs a voltage between 0 and 5V on a pin. The function breaks the range between 0 and 5V into 255 little steps. Note that we are not turning the LED on to full brightness (255) in this code so that the night-light is not too bright. Feel free to change these values and see what happens.

NESTED IF STATEMENTS:

```
if(logic statement){
  if(logic statement){

  }
}
```

A nested **if** statement is one or more **if** statements "nested" inside of another **if** statement. If the parent **if** statement is true, then the code looks at each of the nested **if** statements and executes any that are true. If the parent **if** statement is false, then none of the nested statements will execute.

MORE LOGICAL OPERATORS:

```
(potentiometer > 0 &&

  potentiometer <= 150)
```

These **if** statements are checking for two conditions by using the AND **&&** operator. In this line, the **if** statement will only be true if the value of the variable **`potentiometer`** is greater than 0 AND if the value is less than or equal to 150. By using **&&**, the program allows the LED to have many color states.

DEFINING A FUNCTION:

```
void function_name(){
}
```

This is a definition of a simple function. When programmers want to use many lines of code over and over again, they write a function. The code inside the curly brackets "executes" whenever the function is "called" in the main program. Each of the colors for the RGB LED is defined in a function.

CALLING A FUNCTION:

```
function_name();
```

This line "calls" a function that you have created. In a later circuit, you will learn how to make more complicated functions that take data from the main program (these pieces of data are called **parameters**).

CODING CHALLENGES

ADD MORE COLORS: You can create many more colors with the RGB LED. Use the `analogWrite()` function to blend different values of red, green and blue together to make even more colors. You can divide the potentiometer value and make more nested **if** statements so that you can have more colors as you twist the knob.

MULTI-COLOR BLINK: Try using delays and multiple color functions to have your RGB LED change between multiple colors when it is dark.

CHANGE THE THRESHOLD: Try setting your threshold variable by reading the value of a potentiometer. By turning the potentiometer, you can then change the threshold level and adjust your night-light for different rooms.

FADING THE LED: Use `analogWrite()` to get your LED to pulse gently or smoothly transition between colors.

TROUBLESHOOTING

The LED never turns on or off	Open the Serial Monitor and make sure that your photoresistor is returning values between 0 and 1023. Cover the photoresistor; the values should change. If they do not change, check your circuit. Make sure that your threshold variable sits in between the value that the photoresistor reads when it is bright and the value when it is dark (e.g., bright = 850, dark = 600, threshold = 700).
My LED doesn't show the colors that I expect	Make sure that all three of the pins driving your RGB LED are set to `OUTPUT`, using the `pinMode()` command in the setup section of the code. Then make sure that each leg of the LED is wired properly.
Nothing is printing in the Serial Monitor	Try unplugging your USB cable and plugging it back in. In the Arduino IDE, go to **Tools** > **Port**, and select the right port.

You've completed Circuit 1D!

Continue to Project 2 to explore using buzzers to make sound.