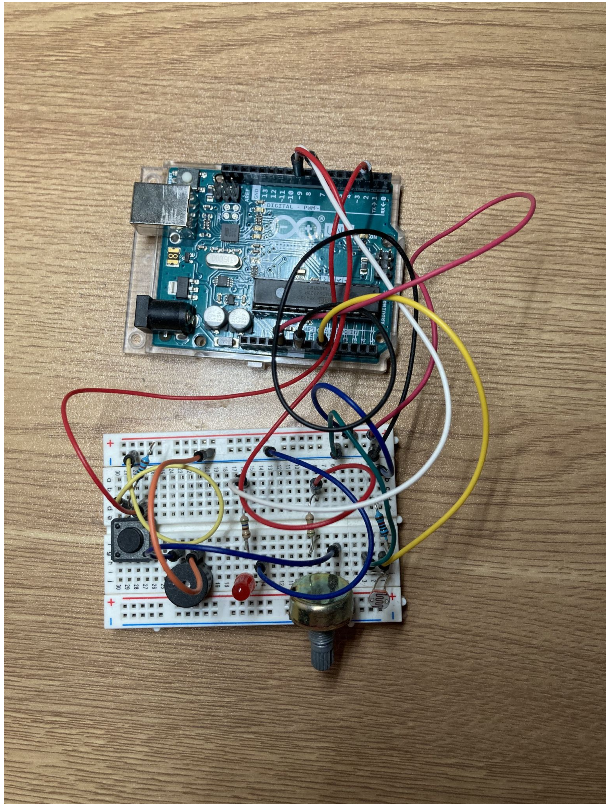
# Embedded System Workshops

08. Photoresistors

*CCA Girls Who Code*

# Project Overview

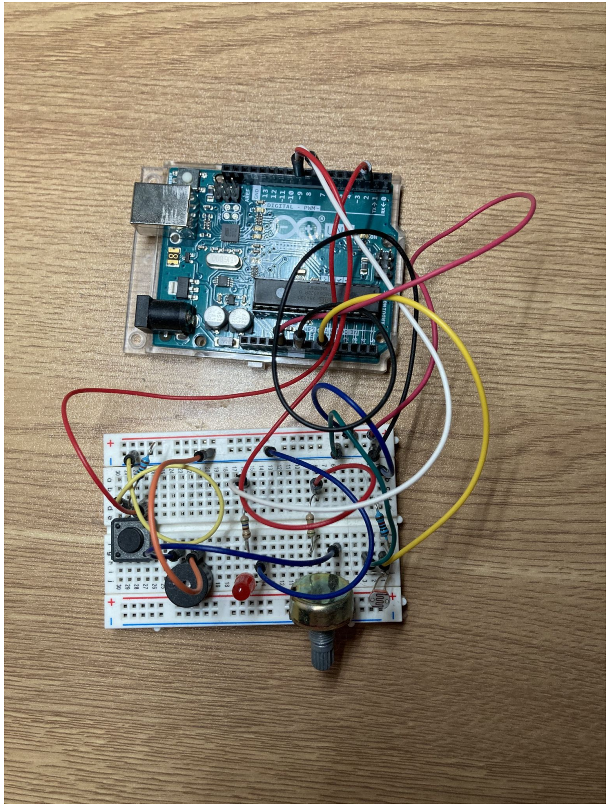
- Purpose
  - Learn about photoresistors and use them to create a box that will play music when opened
  - Combine skills and parts used in previous projects
- Grab your kit, and let’s get started!



# What are we making?

→ "Rick-Roll" Box

- When a box is opened, a photoresistor trigger will cause a piezo buzzer to play Rick Astley's "Never Gonna Give You Up"
- Based off Samantha Lagestee's Rickroll Box



# Parts List

- Arduino UNO R3 Controller Board
- USB Cable
- Breadboard
- Active Buzzer
- Photoresistor
- LED
- Potentiometer
- Button
- Resistor, 10K ohm (x2)
- Resistor, 560 ohm (x2)
- Male-to-male jumper wires
- Cardboard box (optional)

# Project

# Types of I/O in this project

Inputs

- Photoresistor
  - Determines when the box is opened or closed
- Potentiometer
  - Adjusts the volume of the buzzer
- Button/Switches
  - Change speed of music

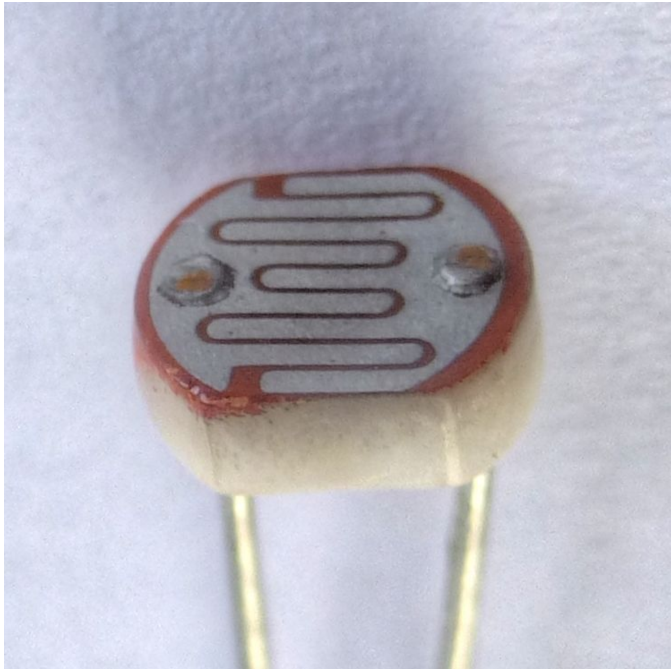
Outputs

- Piezo buzzer
  - Plays music
- LED
  - Blinks in time with the notes
- Serial output (monitor)
  - Outputs the lyrics of the song to the monitor

**Review Q:** What kind of inputs are these? Outputs? Are they digital or analog?

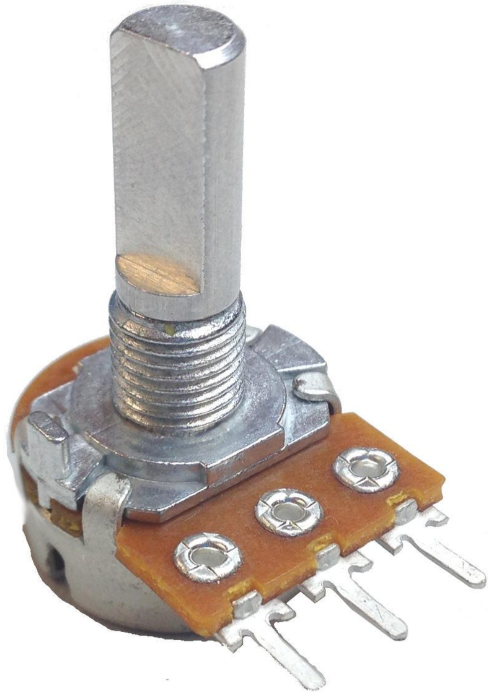
# Photoresistors

A photoresistor is a type of resistor whose resistance decreases as the brightness of the light hitting it increases. This makes it useful as a way to detect the amount of light in an area, perhaps to turn on certain devices above a certain brightness level.



# Review: What is a Potentiometer?

- A potentiometer is essentially a variable resistor: you can change how much resistance is applied by twisting the knob
- This is useful for adjusting brightness, power, etc.



# Review: What are Switches?

A switch is a device that you can use to open or close a circuit at will. A closed circuit allows current to flow through it. An open circuit has a gap in the circuit, preventing current from flowing through it.

**Why is this relevant?**

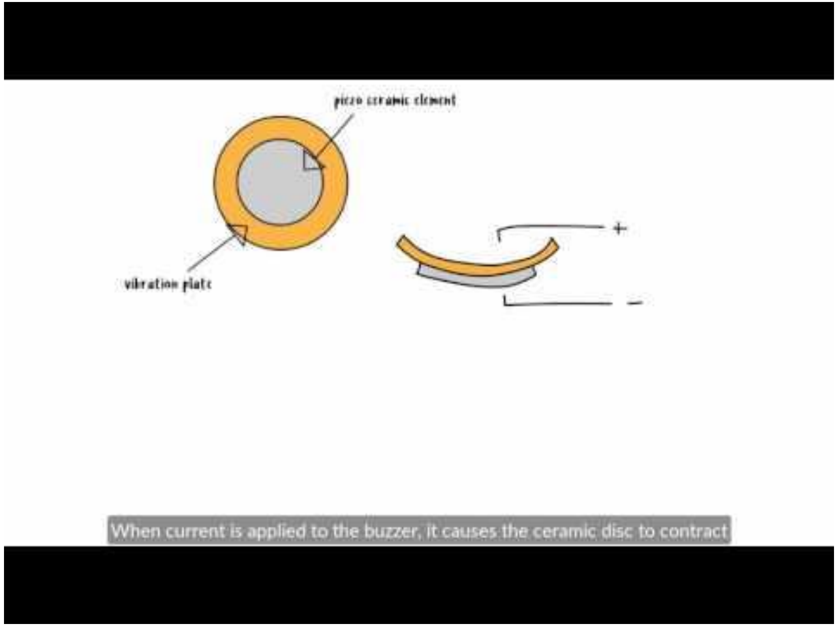
A button is a type of switch that closes the circuit when pressed down and opens it when released.

A button

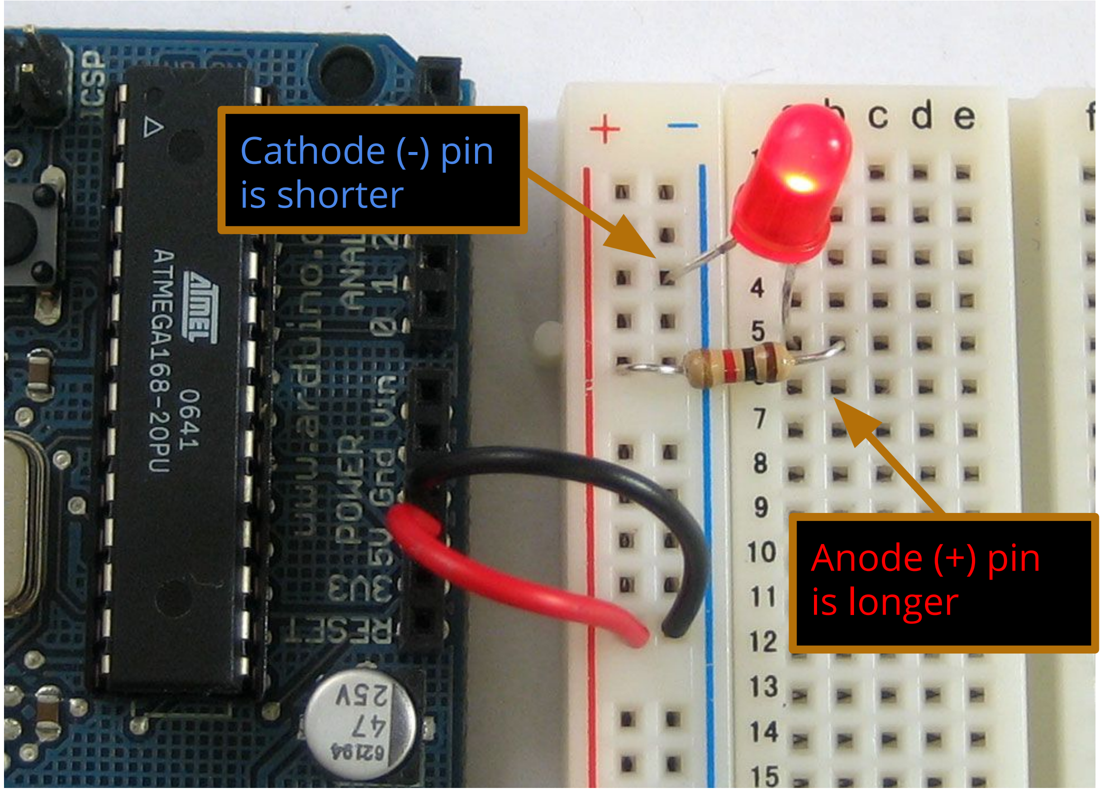
# Buzzers

A buzzer is designed to play a specific tone when given an input. Different frequencies correspond to different tones being played.

When current is applied to the buzzer, an internal ceramic disk contracts or expands against a surrounding disk, creating sound.



# Review: LEDs




NOTE: Make sure the power input is connected to the Anode, and the ground pin is connected to the Cathode. Make sure you also have a resistor between either the power input and Anode, or the Cathode and ground pin. Failing to do either of these things can damage the LED or the Arduino.

# Serial Monitor

The Serial Monitor allows you to output values and write them to a place where you can observe how the values change in real time.

To begin a serial monitor, you must enter a line in the void setup titled “Serial.begin([baud rate]);”. The baud rate we will be using is 9600, making that line “Serial.begin(9600);”

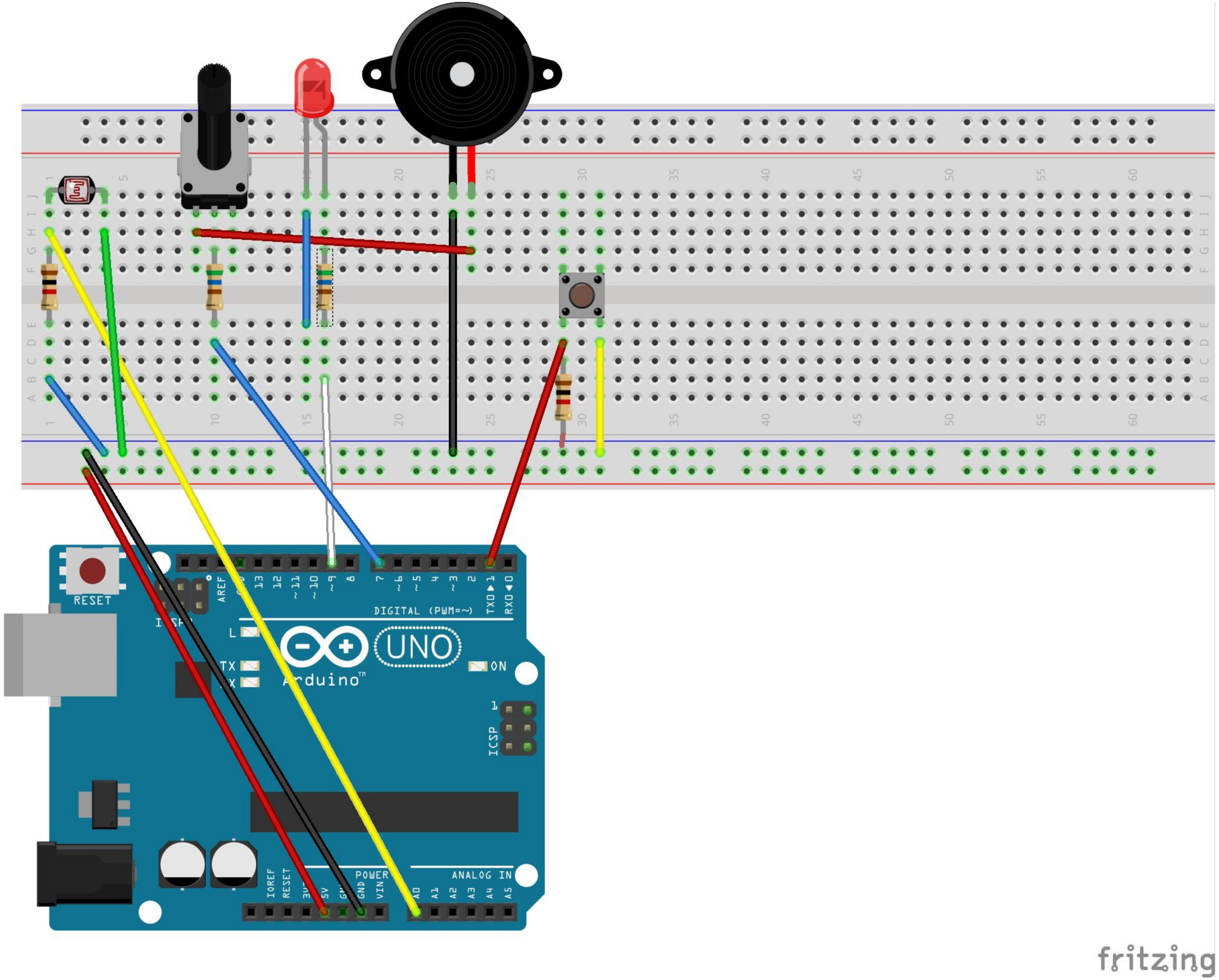
“Serial.print(value);” prints the value to the monitor.

“Serial.println(value);” prints the value to a new line on the monitor

# Schematic

10k ohm resistors connected to photoresistor and button

560 ohm resistors connected to potentiometer and LED

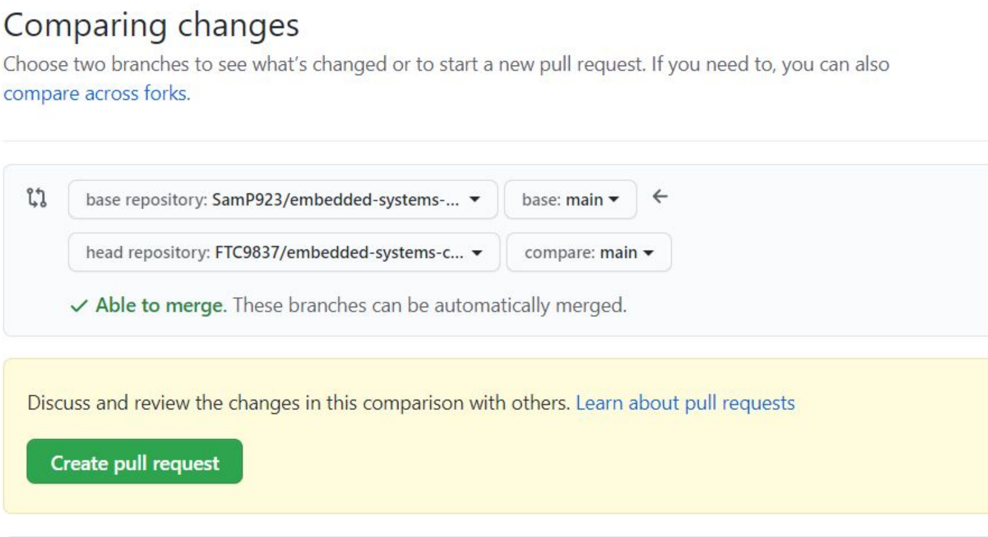


# Grab the Starter Code from GitHub

If you haven't made the repository yet, check these slides

- → Go to your repository (username/embedded-systems-course)
- → Click "Compare"

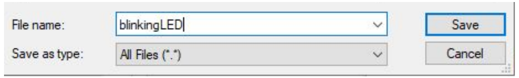

- → Switch the repos so yours is the base repository and FTC9837 is the head

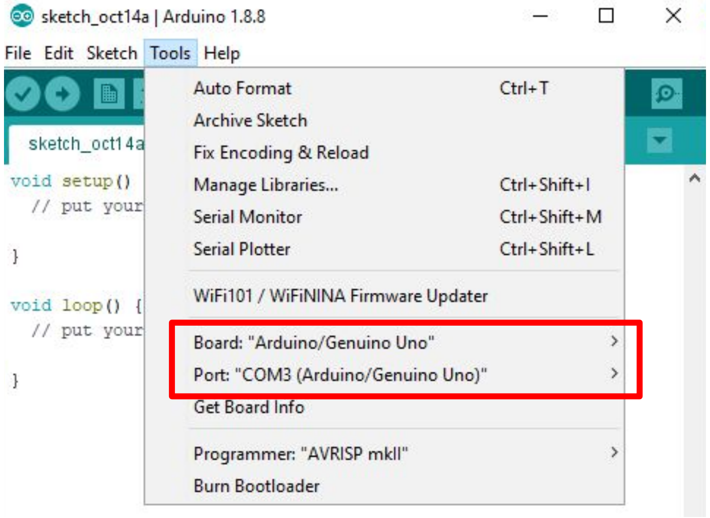



- → Create pull request (x2)
- → Merge pull request (and confirm)
- → You should have the lesson8 folder in your repository

# Review: Setting up Arduino

- Find and open Arduino on your desktop
- Click “File” in the top left corner and click save
- Save this tab as “rickroll”
- Connect the USB cord in your kit to the Arduino and the computer (USB port is on the left side of the monitor)
- Open the “Tools” Window and make sure the board has been recognized and the port is “COM#(Arduino/Genuino Uno)”

# Rick Roll Code

View the fullcode on GitHub!

RickRollBoxCode

```
#define  a3f    208     // 208 Hz
#define  b3f    233     // 233 Hz
#define  b3     247     // 247 Hz
#define  c4     261     // 261 Hz MIDDLE C
#define  c4s    277     // 277 Hz
#define  e4f    311     // 311 Hz
#define  f4     349     // 349 Hz
#define  a4f    415     // 415 Hz
#define  b4f    466     // 466 Hz
#define  b4     493     // 493 Hz
#define  c5     523     // 523 Hz
#define  c5s    554     // 554 Hz
#define  e5f    622     // 622 Hz
#define  f5     698     // 698 Hz
#define  f5s    740     // 740 Hz
#define  a5f    831     // 831 Hz

#define rest    -1

int piezo = 7; // passive buzzer pin
```

```
int piezo = 7; // passive buzzer pin
int led = 9; // LED pin
int button = 2; // button pin
int sensor = A0; // photoresistor pin

volatile int beatlength = 100; // determines tempo
float beatseparationconstant = 0.3;

int threshold;

int a; // part index
int b; // song index
int c; // lyric index

boolean flag;

// Parts 1 and 2 (Intro)

int song1_intro_melody[] =
{c5s, e5f, e5f, f5, a5f, f5s, f5, e5f, c5s, e5f, rest, a4f, a4f};

int song1_intro_rhythmn[] =
{6, 10, 6, 6, 1, 1, 1, 1, 6, 10, 4, 2, 10};

// Parts 3 or 5 (Verse 1)

int song1_verse1_melody[] =
{ rest, c4s, c4s, c4s, c4s, e4f, rest, c4, b3f, a3f,
  rest, b3f, b3f, c4, c4s, a3f, a4f, a4f, e4f,
  rest, b3f, b3f, c4, c4s, b3f, c4s, e4f, rest, c4, b3f, b3f, a3f,
```

To test your code, click the checkmark then the arrow!

# Thank you!

CCA Girls Who Code
ccagirlswhocode@gmail.com
www.ccagirlswhocode.weebly.com

# Production Team

Curriculum Director: Stefan Prestrelski

Teaching Assistants: Sarah Luo, Samantha Prestrelski