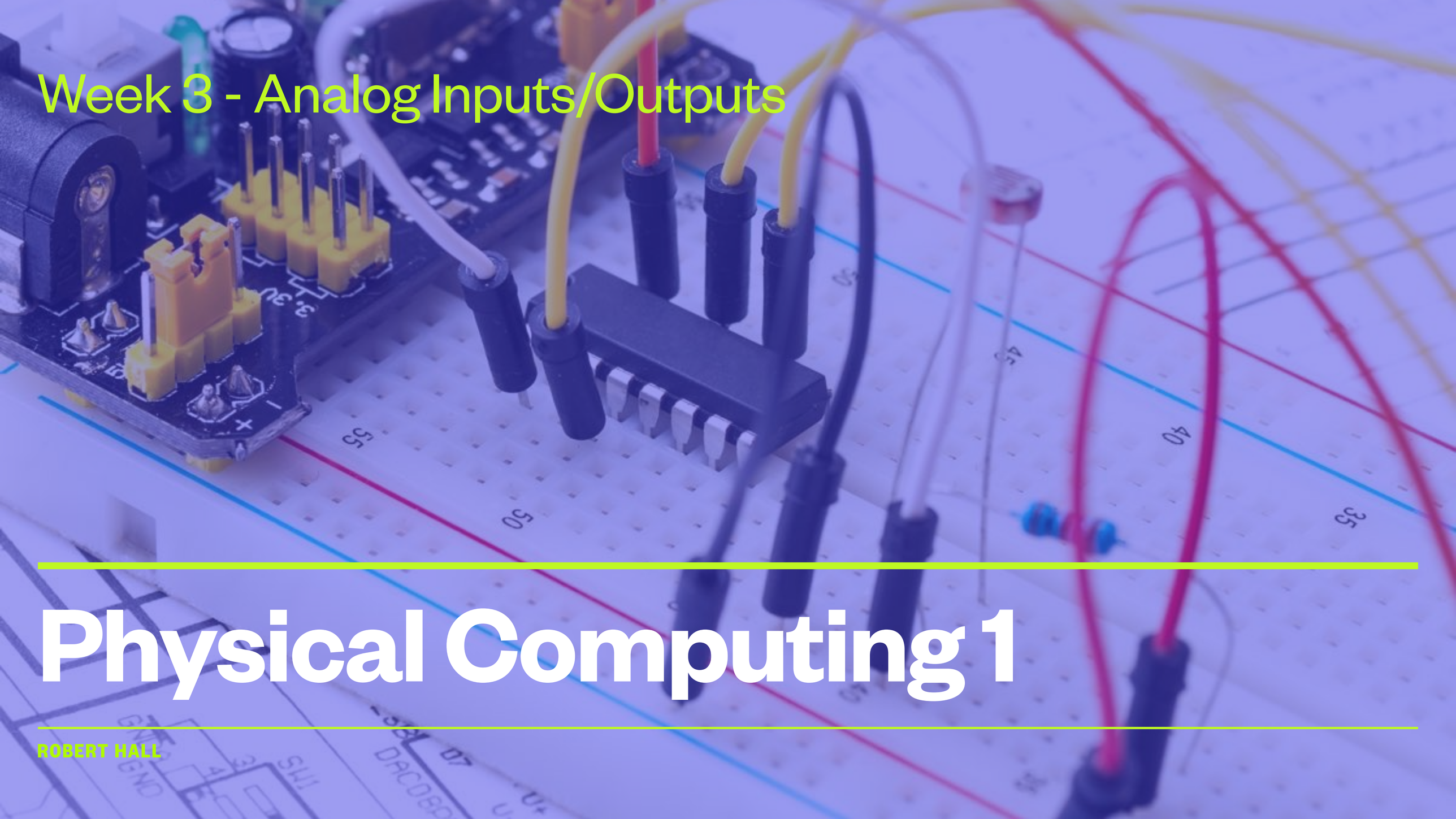


Week 3 - Analog Inputs/Outputs

Physical Computing 1

ROBERT HALL

Any Questions so far?

Homework Review

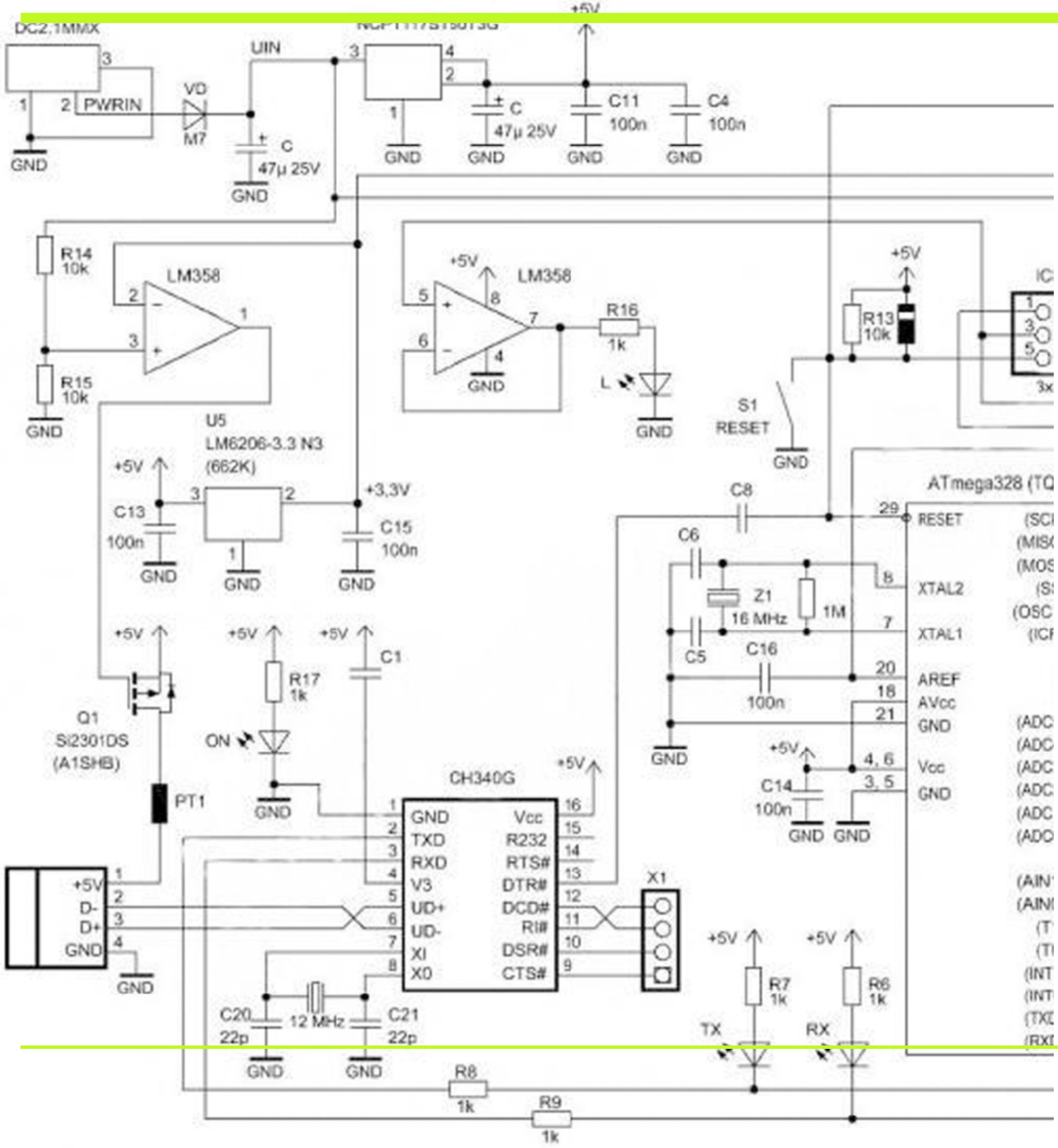
Inspiration Time!

How was this done?

2014/11

Plan for Today

- Homework Review
- Inspiration Time
- Recap of Last Week
- Analog Inputs and Outputs
- LDRs
- Servo Motors
- RGB LEDs
- Idea Generation
- 2nd Assignment



Schematics

- A map of the connections between the components in your project.
- Components are represented by Symbols.
- Wires/connections represented by straight lines.

Action

Statement :
An individual instruction, usually a function call or setting a variable to a number.

```
int ledLevel = 8;

analogWrite(ledLevel);
```

Action

Functions :
Re-usable code blocks. Take arguments as inputs that help them know what to do.

```
void setup() {
 pinMode( 1 , OUTPUT ) ;
 delay(200);
}
```

Memory

Variables :
Ways of storing and keeping track of values (numbers, text, anything else!)

You can think of them as folders that you can put information in.

Have a specific TYPE unlike in P5.js

Memory

Variables in Arduino look different from what you might be used to from P5.js

In P5.js you might say “`var number = 5;`”

In Arduino you have to specify what TYPE of variable you have.
“`int number = 5;`”

Memory

The most common TYPES are:

`int` - numbers with no decimal point
`float` - numbers with a decimal point
`string` - words and sentences made of letters

Decisions:
We make decisions primarily with ‘if’ statements.

```
if(resistorLevel > 500){
  digitalWrite(LED_BUILTIN, HIGH);
}else{
  digitalWrite(LED_BUILTIN, LOW);
}
```

Any Questions so far?

New this week

Repetition

Loops:
Let us repeat an action a particular number of times.

Loops on Arduino are very similar to P5.js

```
for(int i = 0; i<100; i++){
  Serial.println(i);
}
```

Try This :

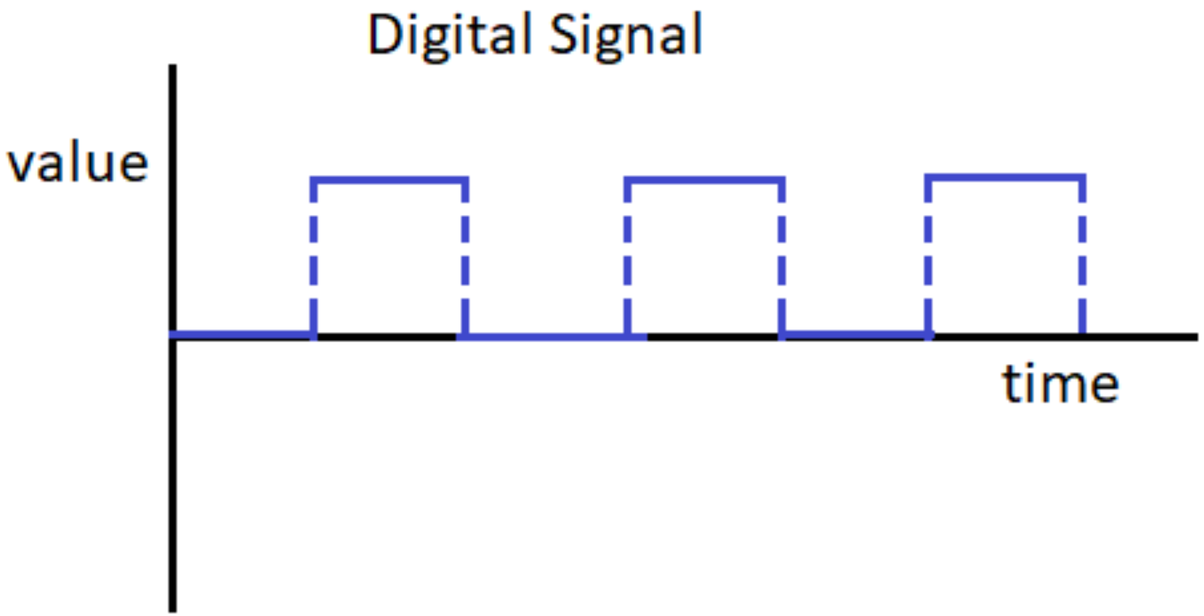
Use a loop to write the numbers from 1-10 in the Serial Monitor

Try This:

```
void setup(){
 for(int i = 0; i<10; i++){
  Serial.println(i);
 }
}

void loop(){
}
```

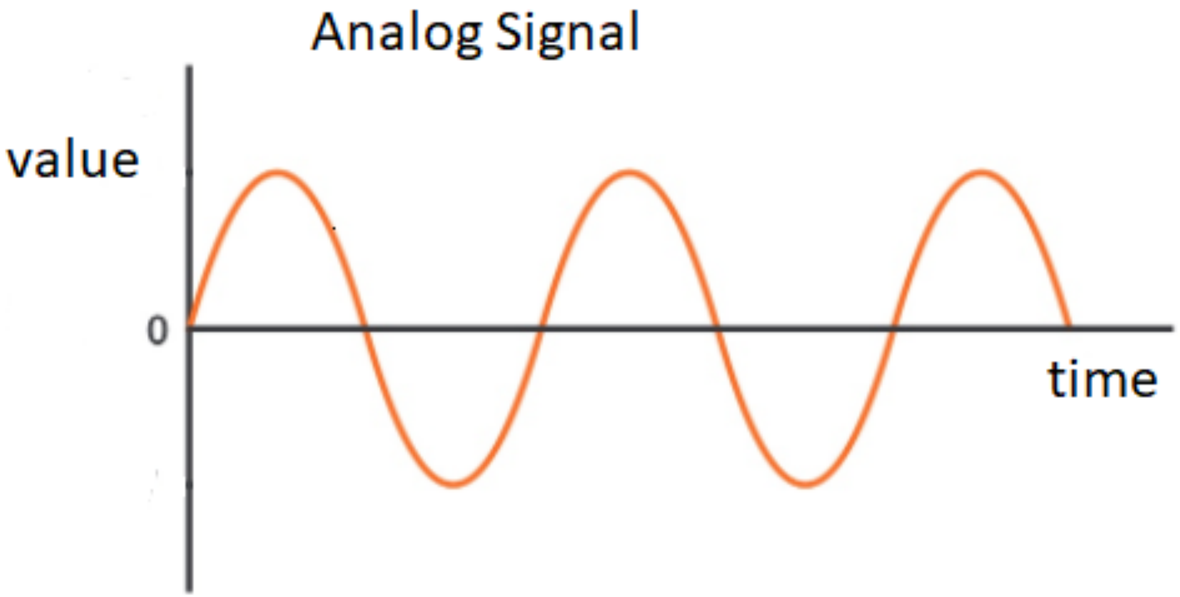
Analog Vs Digital

Digital



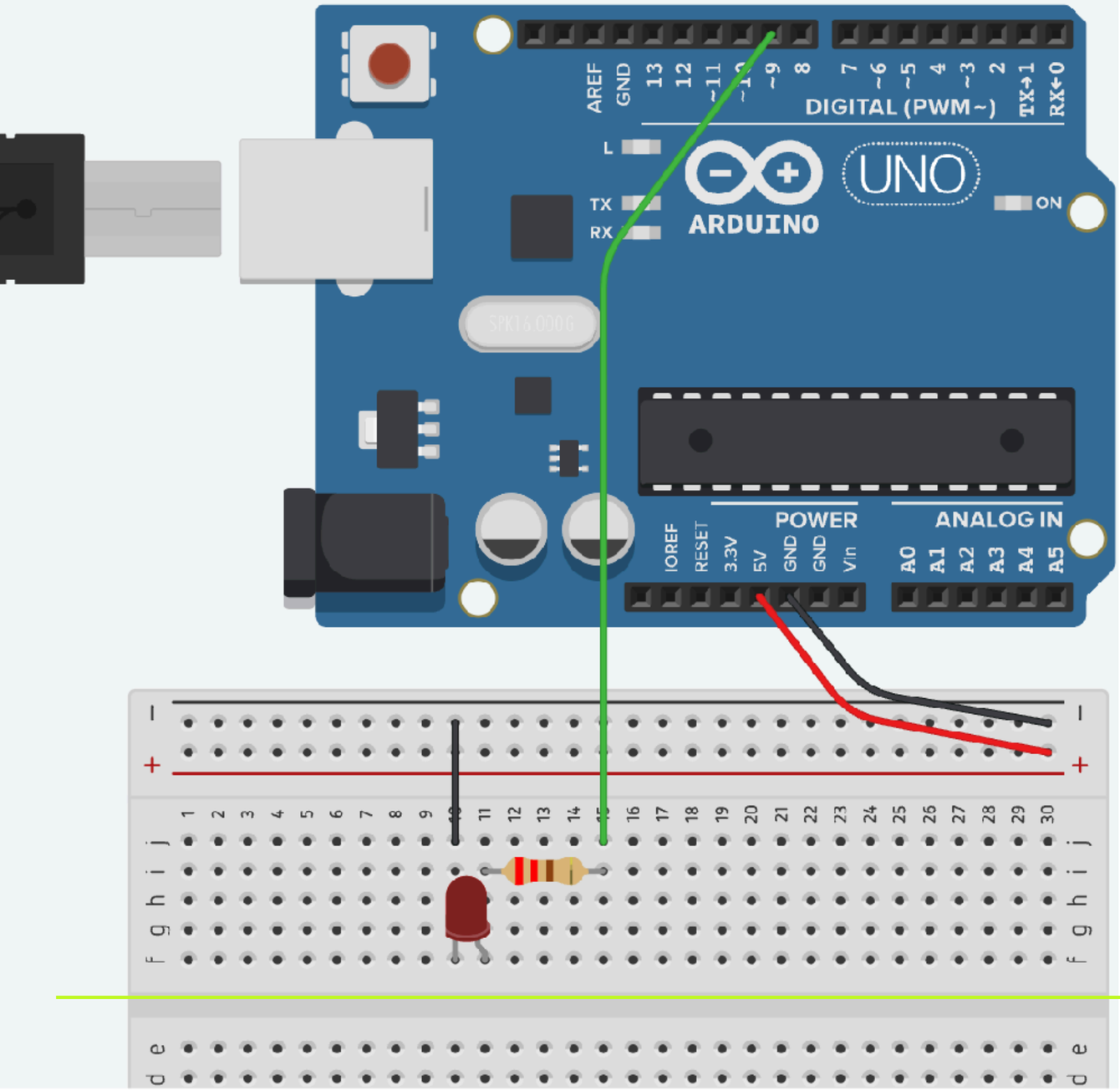
- Digital signals are either ON or OFF
- Digital signals are less susceptible to interference.

Analog



- Analog Signals can be any value, and can move smoothly between values
- Analog Signals are prone to interference as any changes to the signal immediately change the data transmitted

What else do we want to do with LEDs ?

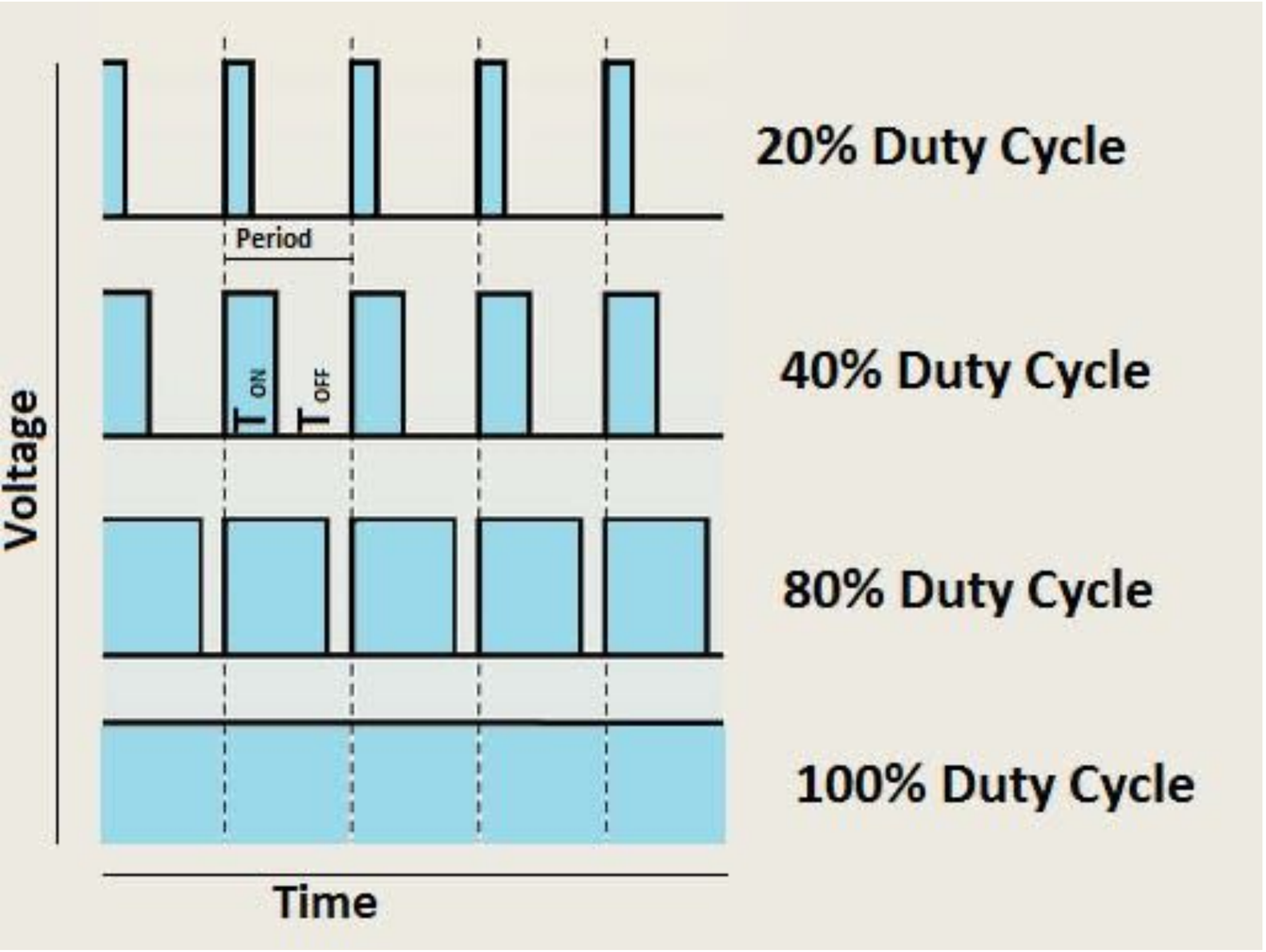


Fade an LED

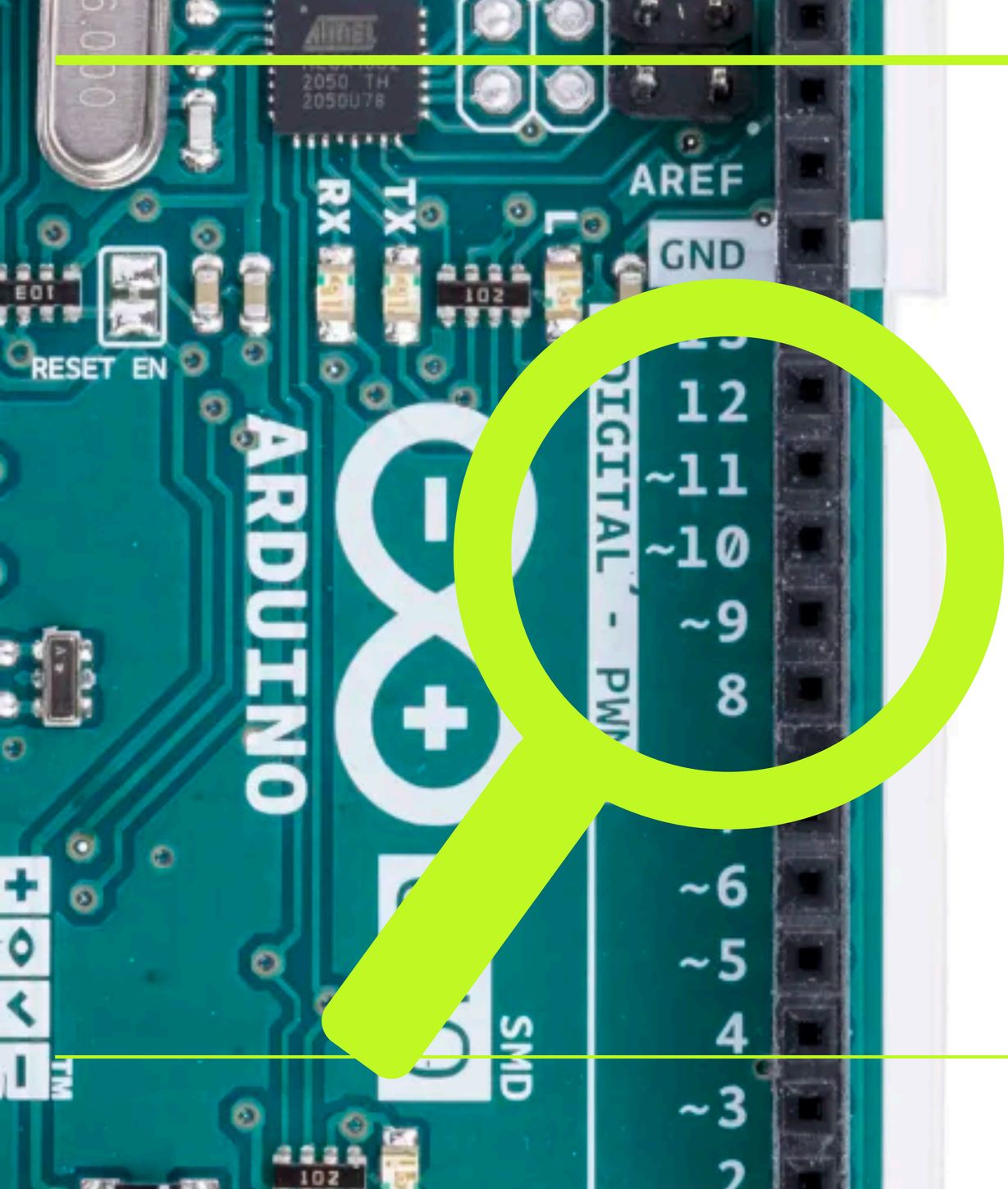
- We can use one of our PWM Pins to fade our LED
- The Pins with PWM have a ~ symbol

What is PWM?

Pulse Width Modulation (PWM)



- PWM is a way of emulating an analog signal by selectively turning on and off a digital signal.
- We can use this signal to dim an LED among other things.
- We call the percentage of time the Digital Signal is ON, the Duty Cycle.



PWM (~) Pins

- The Arduino Uno has 6 PWM Pins on :
- 3,5,6,9,10 and 11
- The Arduino Mega has 15 PWM Pins on :
- 2 - 13, 44 - 46

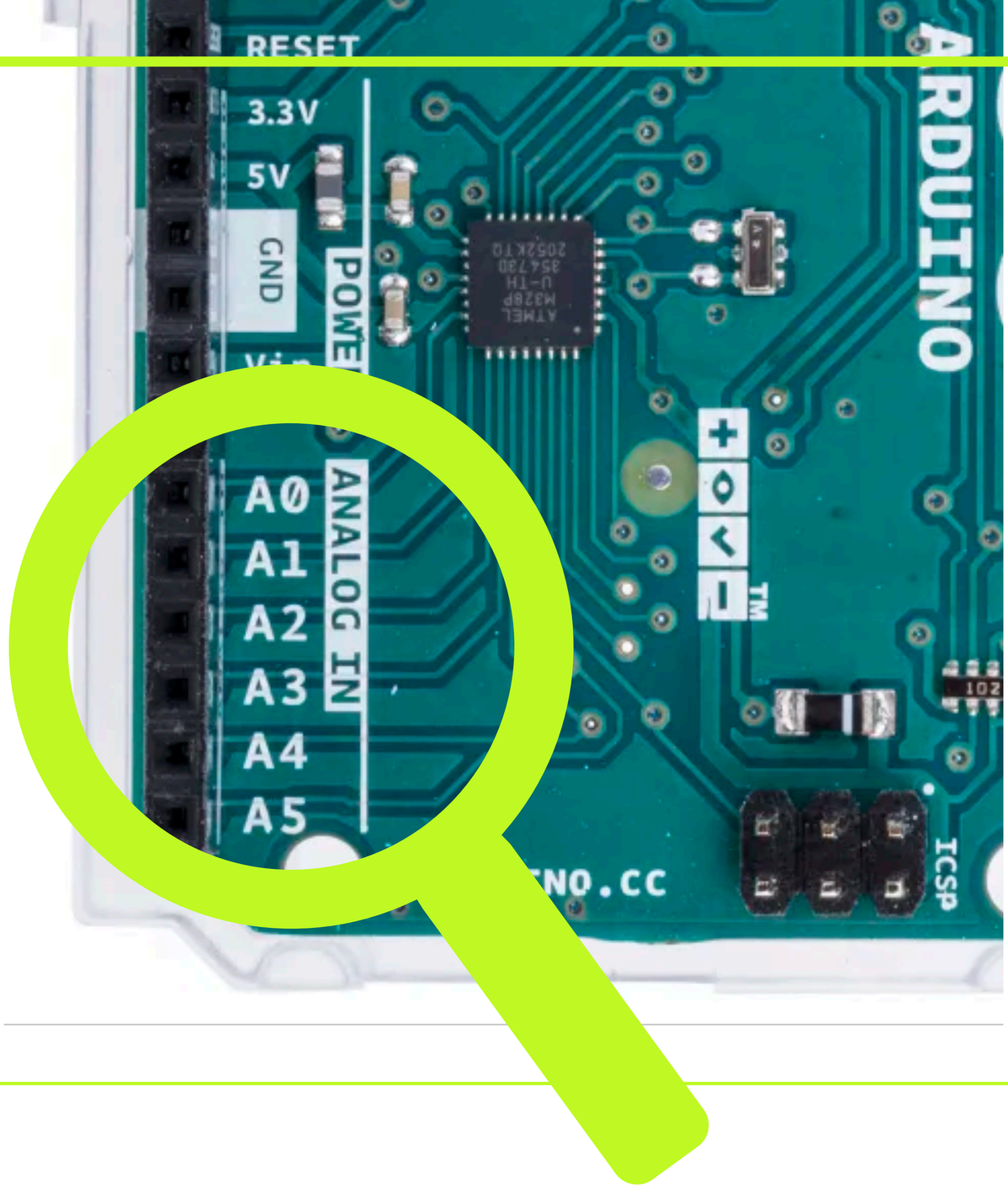
Fade

```
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over a
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading
  if (brightness <= 0 || brightness >= 2
    fadeAmount = -fadeAmount;
  }
```

analogWrite()

- The analogWrite() function takes 2 arguments, the number of the LED pin, and the brightness.
- LED Pin can be the number of any of the PWM Pins
- Brightness can be an integer between 0 and 255
- We don not need to use pinMode with analogWrite().

We also want to get Analog input



Analog Pins

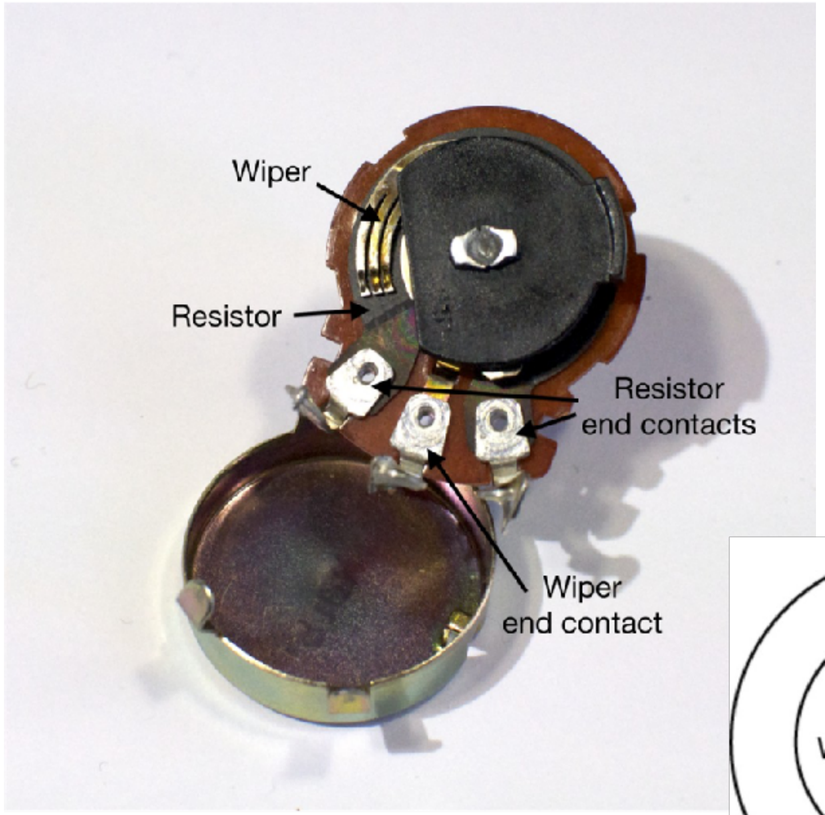
- We can get Analog Inputs from the Analog Pins
- These pins are only inputs but can be read as Digital or Analog.
- We can connect a Potentiometer (Variable Resistor) to these pins to get an Analog Input.

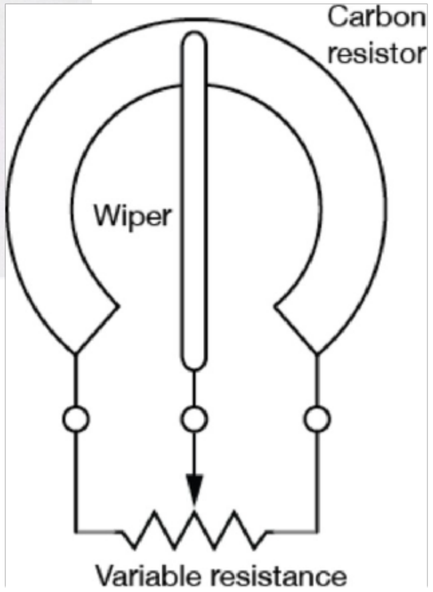
AnalogInput

```
void setup() {
  // declare the ledPin as an OUTPUT:
  pinMode(ledPin, OUTPUT);
}

void loop() {
  // read the value from the sensor:
  sensorValue = analogRead(sensorPin);
  // turn the ledPin on
  digitalWrite(ledPin, HIGH);
  // stop the program for <sensorValue> mi
  delay(sensorValue);
  // turn the ledPin off:
  digitalWrite(ledPin, LOW);
```

analogRead()

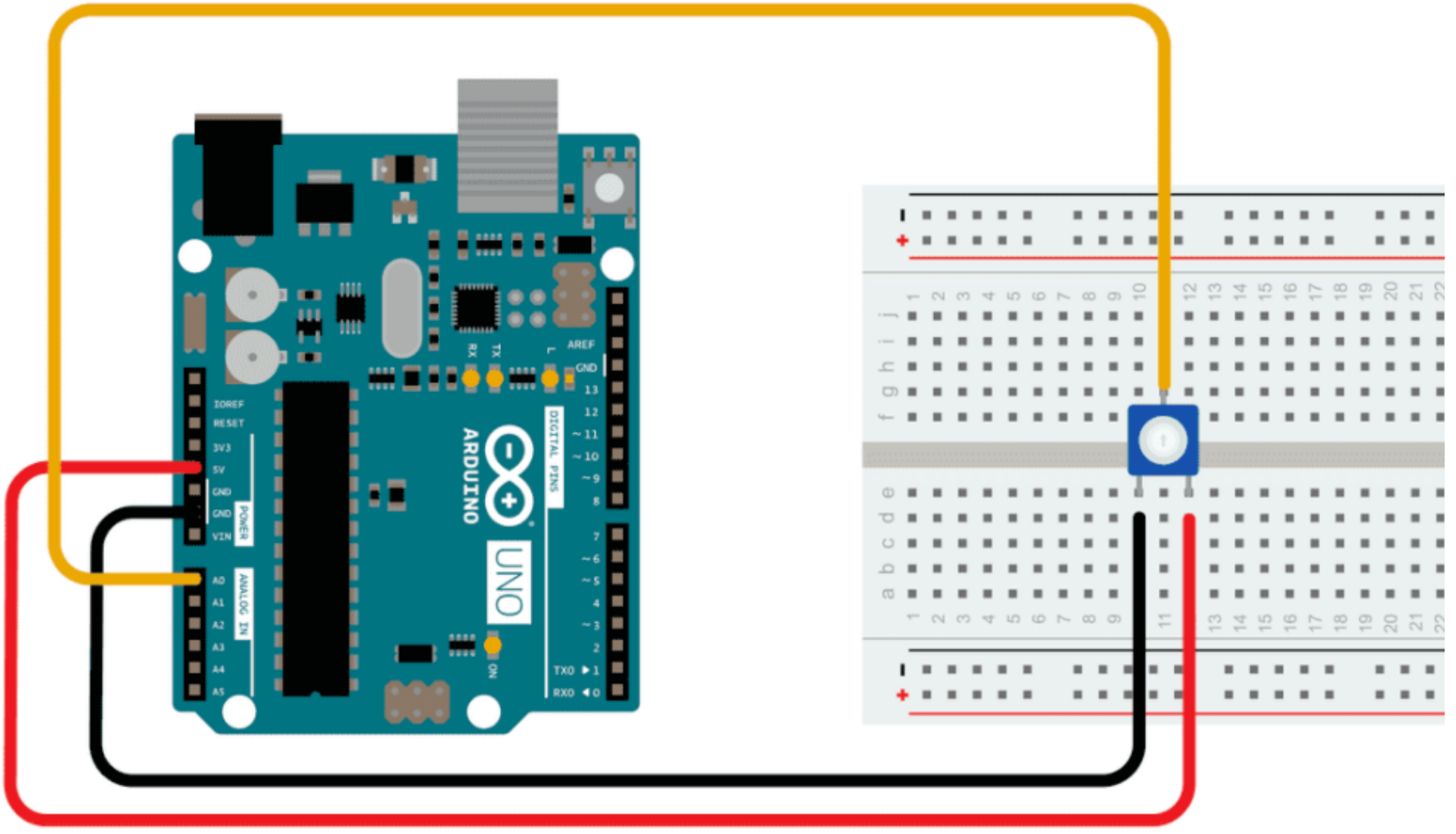
- Just like digitalRead() we use analogRead() to measure what is connected to a Pin.
- Takes the Input Pin number as an Argument
- Returns a number between 0 and 1023
- We do not need to use pinMode with analogRead()

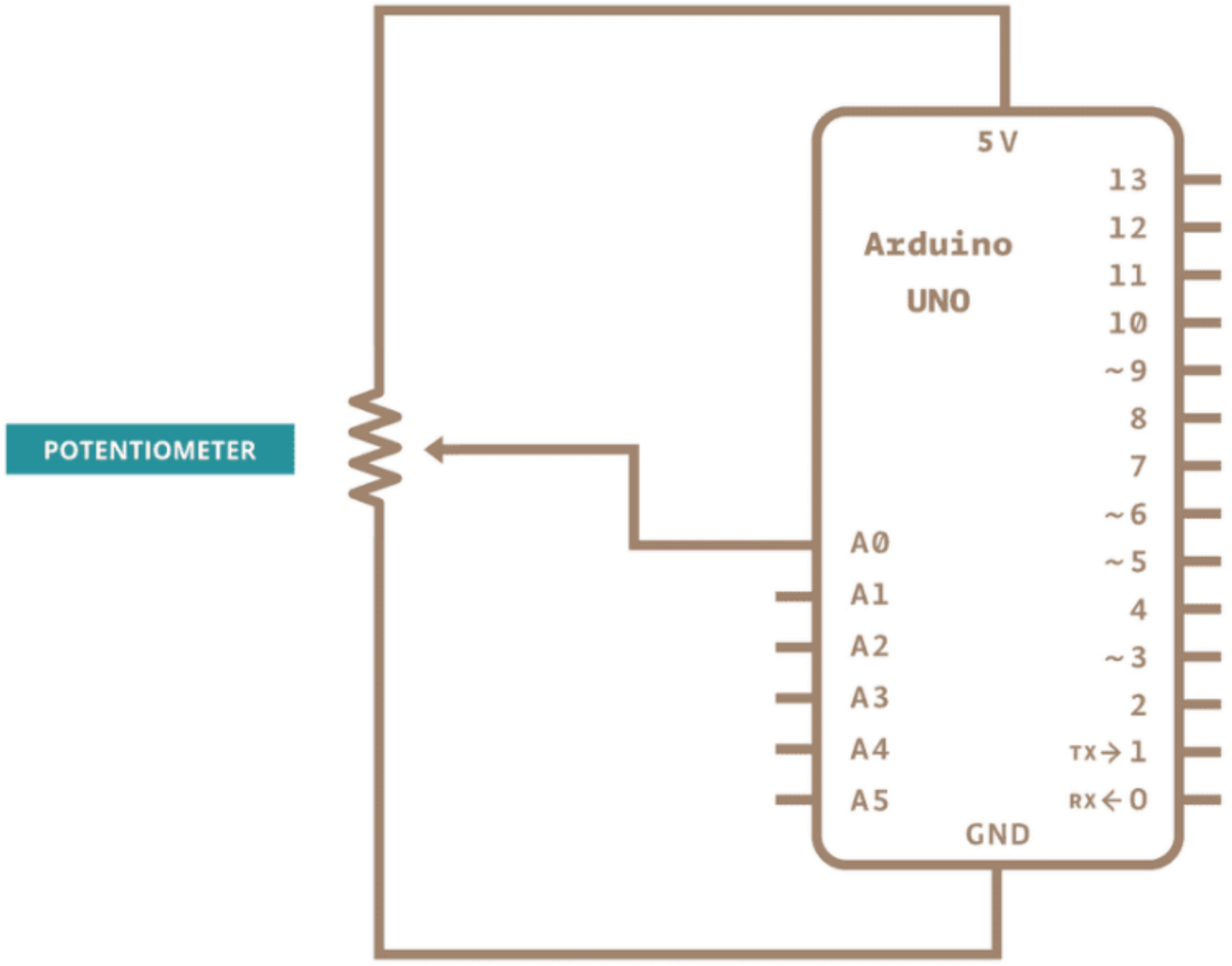


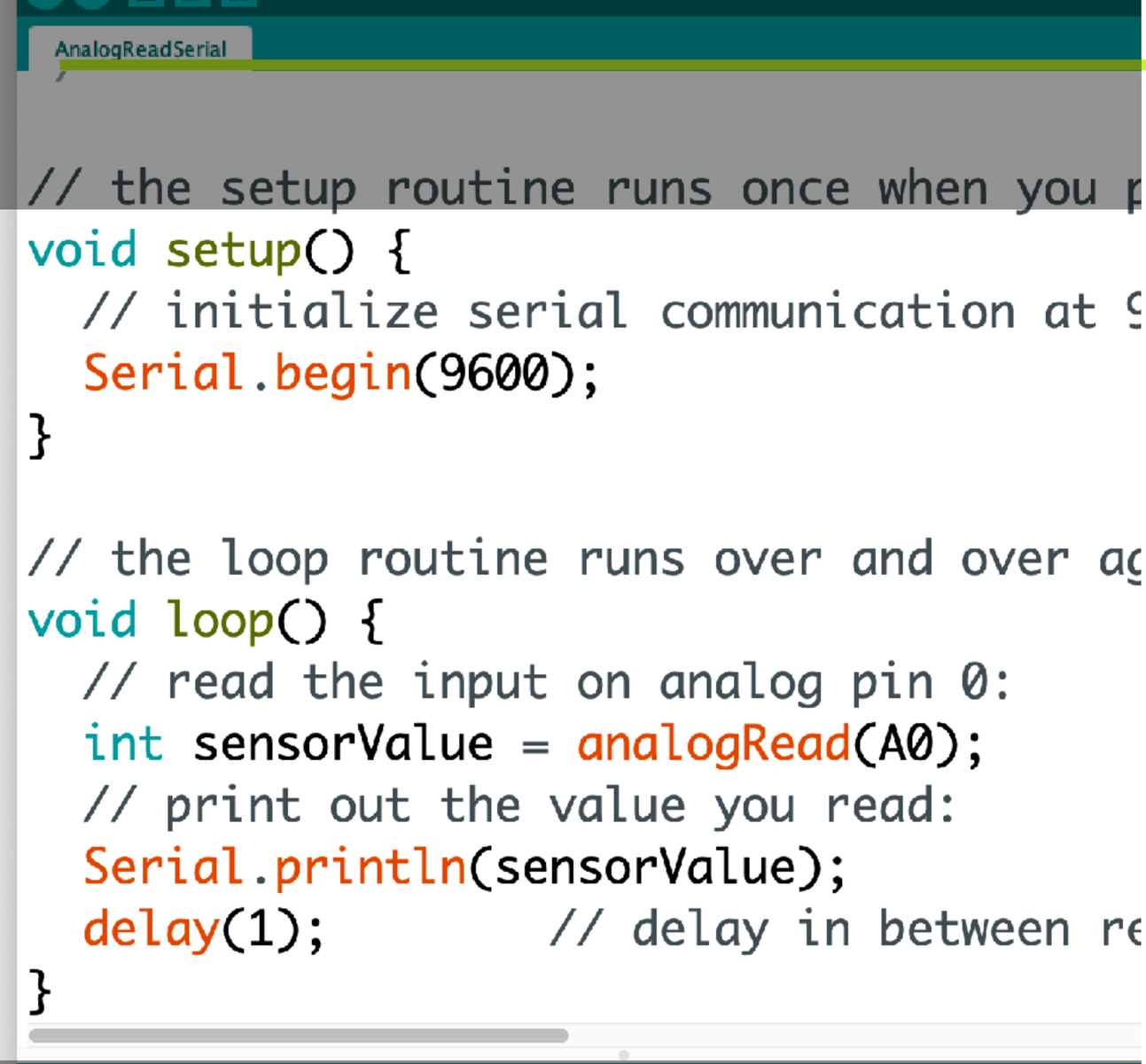


Potentiometer
Variable Resistor

- Potentiometers work by changing how far through a carbon resistor electricity has to travel, the smaller the gap, the less resistance.
- Potentiometers have 3 Pins, GND, 5V and Wiper
- We connect Wiper to the Analog In Pin and the whole device makes a Voltage Divider.


IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
RX
TX
L
AREF
GND
13
12
~ 11
~ 10
~ 9
8
DIGITAL PINS
7
~ 6
~ 5
4
~ 3
2
TX0 1
RX0 0
ON
ARDUINO
UNO


POTENTIOMETER
5V
Arduino
UNO
13
12
11
10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
A0
A1
A2
A3
A4
A5
GND



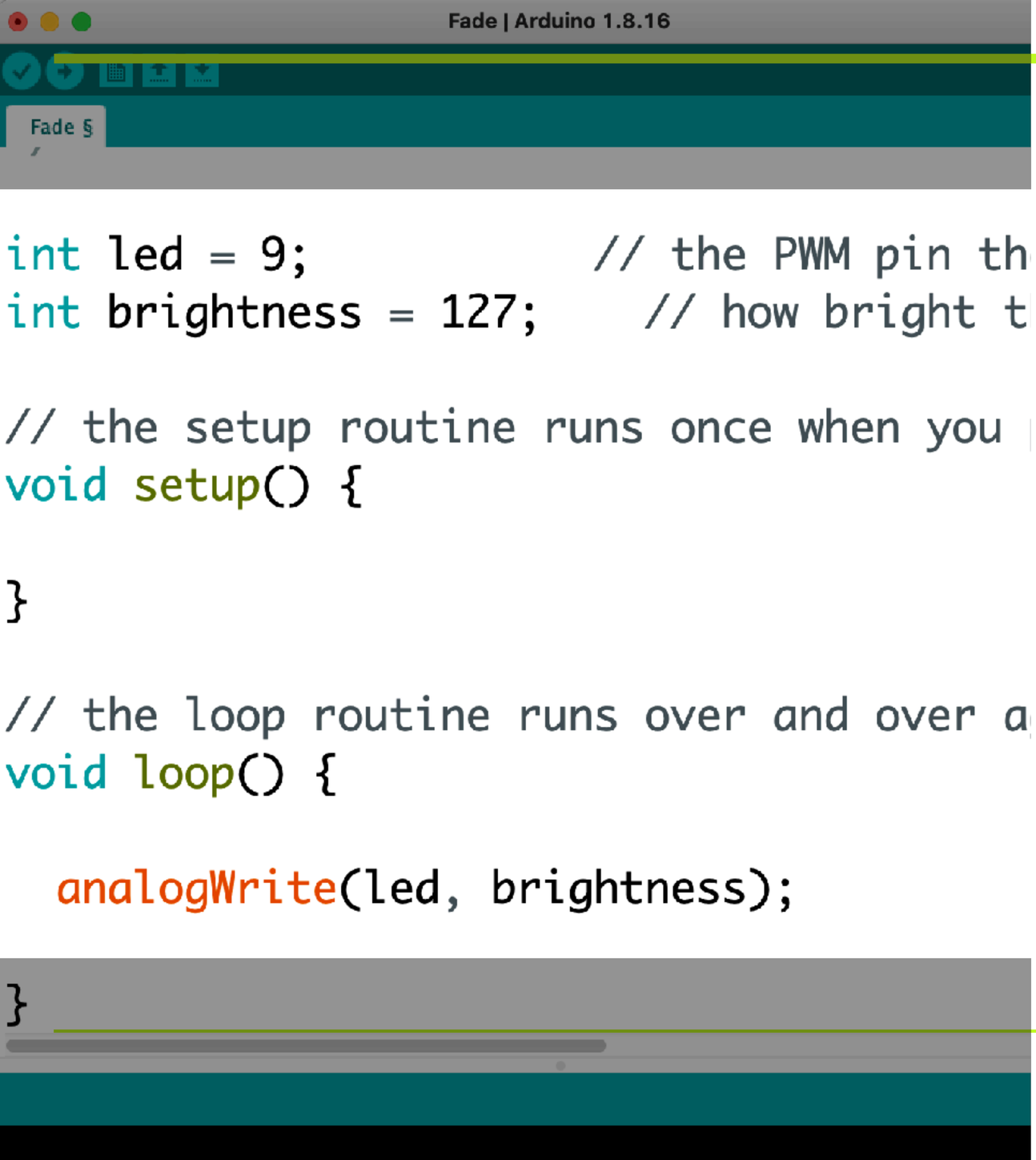
AnalogReadSerial

- Examples>Basics>AnalogReadSerial
- Just like DigitalReadSerial, this code reads the value of our Analog Input and sends it to the Serial Monitor
- Set this Up and see if you can read an Analog Input

There are many ways to do the same thing in code

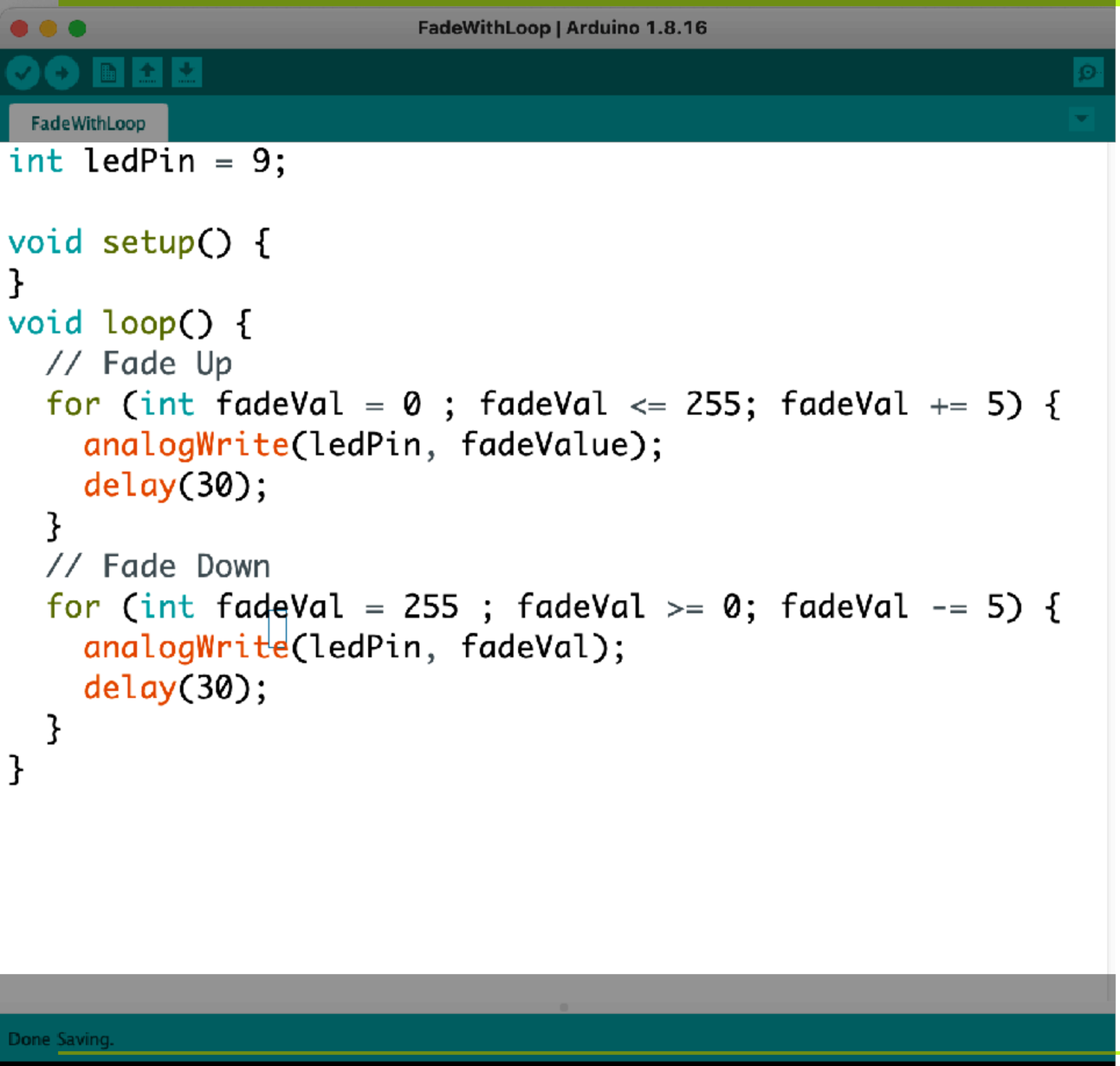
Dimming an LED

- Write a given value > LED Stays at one Brightness
- Fade it with a FOR loop > fades LED, but pauses main loop when it does so
- Fade it with an IF Statement in main loop > does not pause main loop
- Fade it with Incoming Data from a potentiometer > LED can be controlled with Potentiometer



Write Given Value

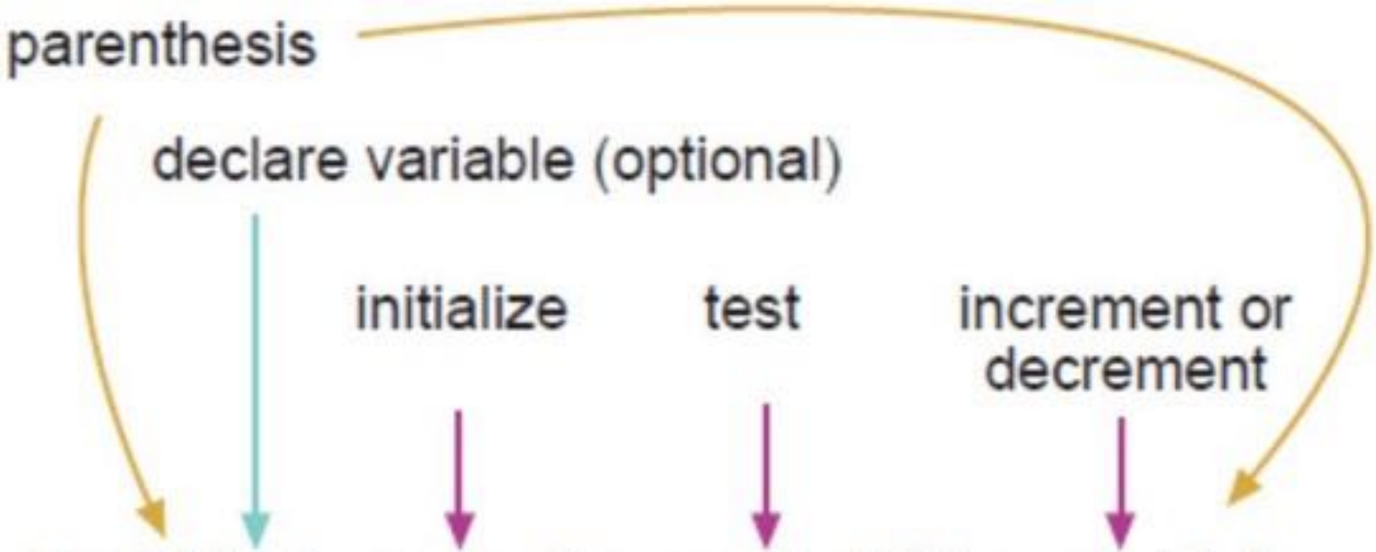
- This code always writes 127 (50% Brightness)
- There are no changing variables, the led is always set to 127



Fade Using Loop

- In this example we use a for loop to count from 0 to 255 and then from 255 to 0
- We use a delay to slow down the loops so the fade is visible and doesn't just look like a flash

For Loop




```
for(int x = 0; x < 100; x++){

   println(x);  // prints 0 to 99
}
```

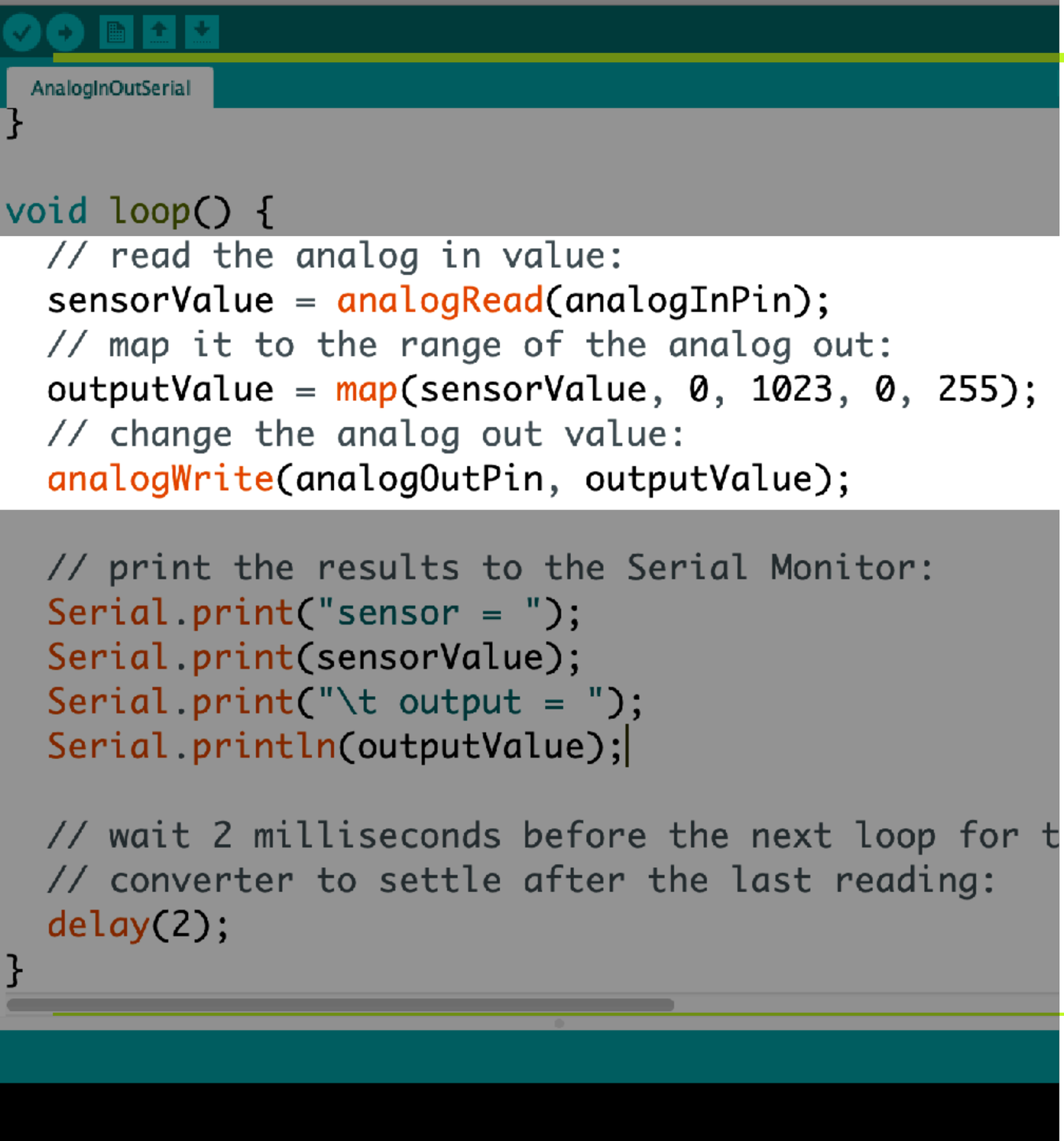
Many ways to increment a variable

- i = i + 1
- i++
- i += 1
- i += otherValue

```
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again fo
void loop() {
  // set the brightness of pin 9:
  analogWrite(led, brightness);

  // change the brightness for next time throug
  brightness = brightness + fadeAmount;

  // reverse the direction of the fading at the
  if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
  }
  // wait for 30 milliseconds to see the dimmin
  delay(30);
}
```

Fade With IF Statement

- This example uses an IF statement to check if we should be fading up or down.
- We use || to mean OR and && to mean AND in IF statements.
- This works by adding fadeAmount to brightness every loop
- The IF statement detects if the brightness value is at 255 or 0 and flips the direction by making fadeAmount positive or negative.



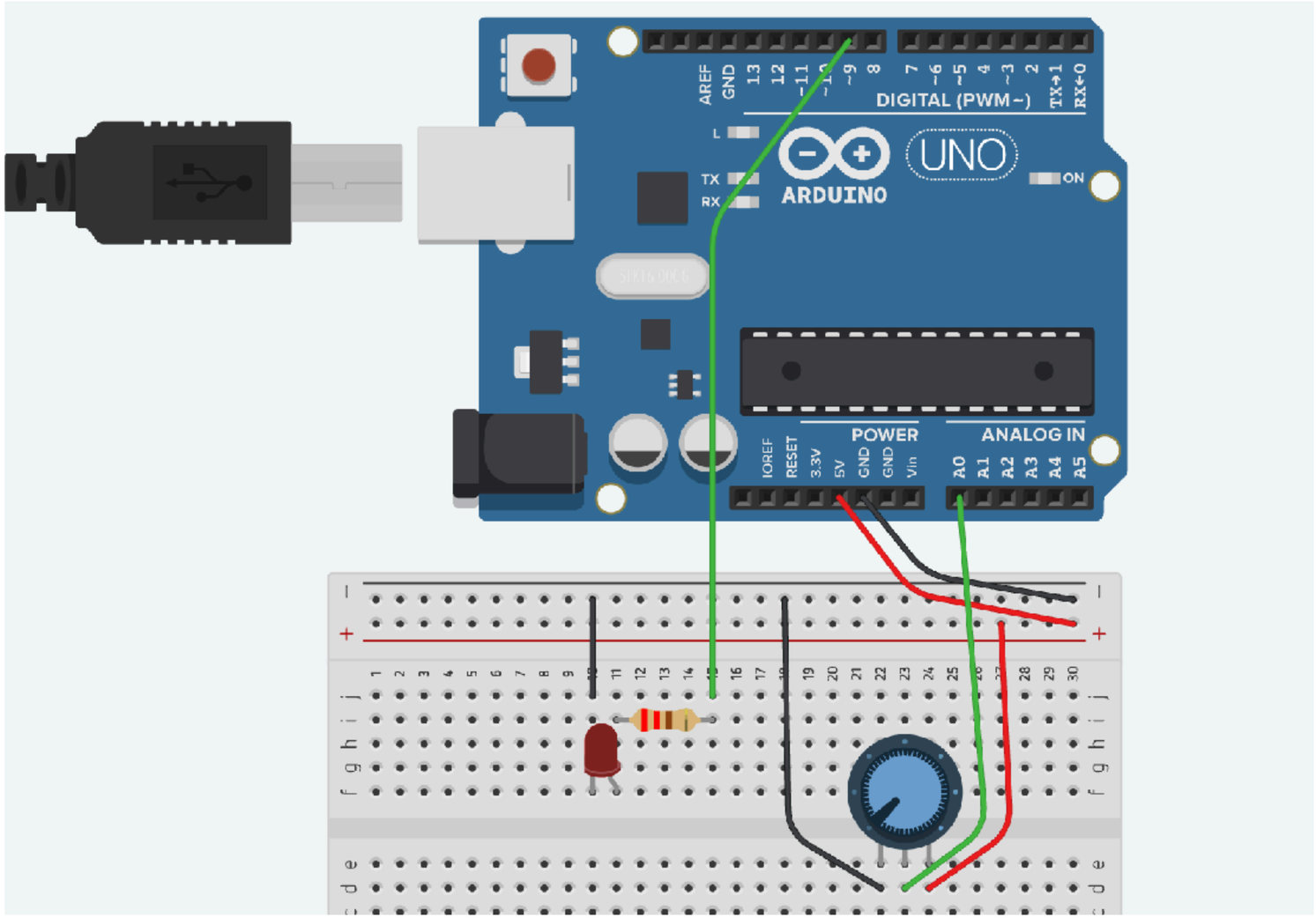
Analog Input and Output

- In this example we use map() to convert a range of values from 0-1023 to 0-255.
- The sensor is read with analogRead()
- The value is mapped with map()
- The LED is set to that level using analogWrite()

map() function

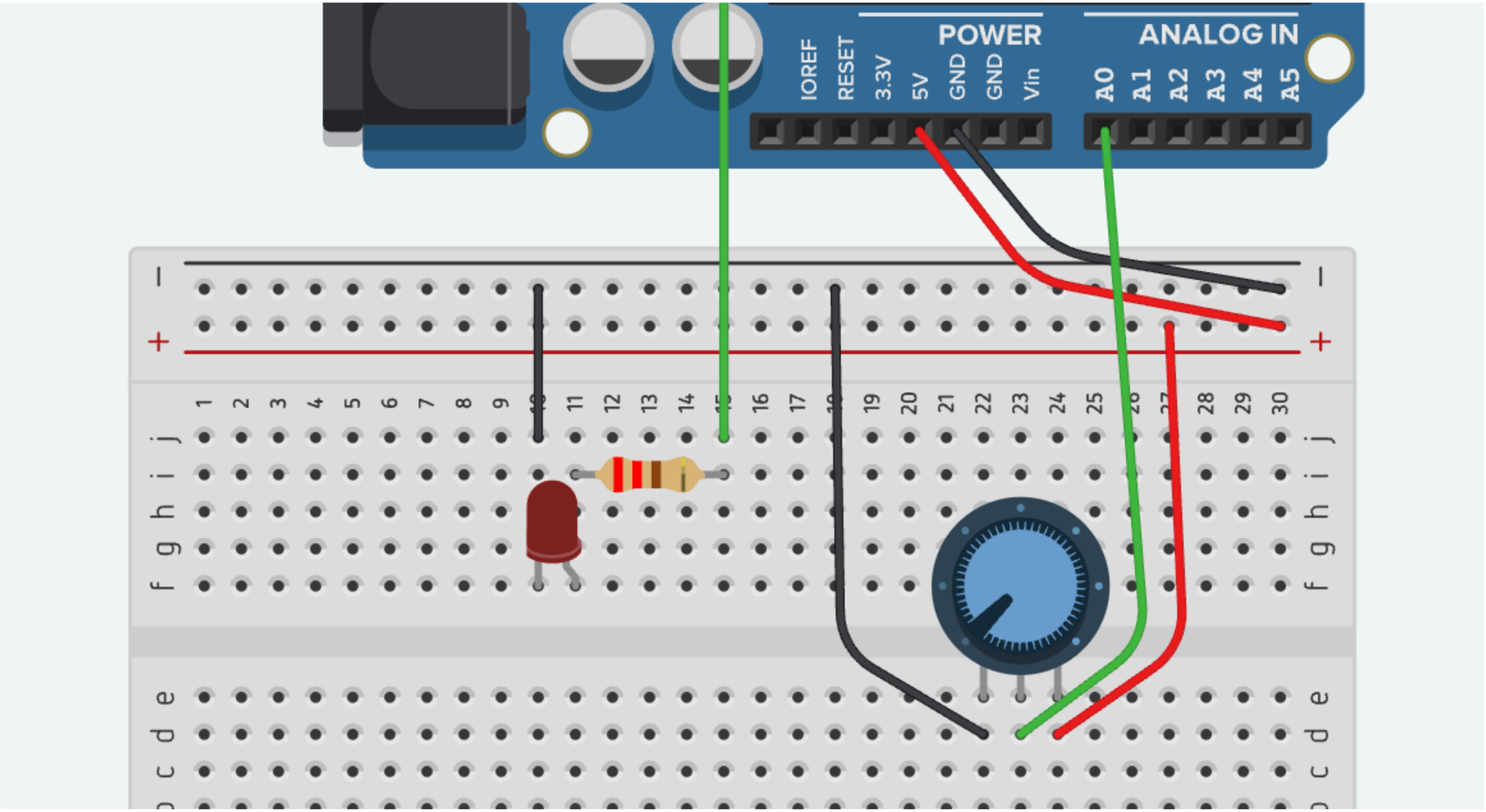
- output = map(input, inputLow, inputHigh, outputLow, outputHigh)

Closeup


AREF
GND
DIGITAL (PWM~)
TX→1
RX←0
L
TX
RX
ARDUINO
UNO
ON
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

Even Closer Up

/\ Up to Pin 9



Practical Exercises

Try out the following examples (the circuits to set up are described in the code) and work out what the code is doing….

- Examples>Analog>AnalogInput
- Examples>Analog>AnalogInOutSerial
- Examples>Basics>ReadAnalogVoltage

Extension Exercises

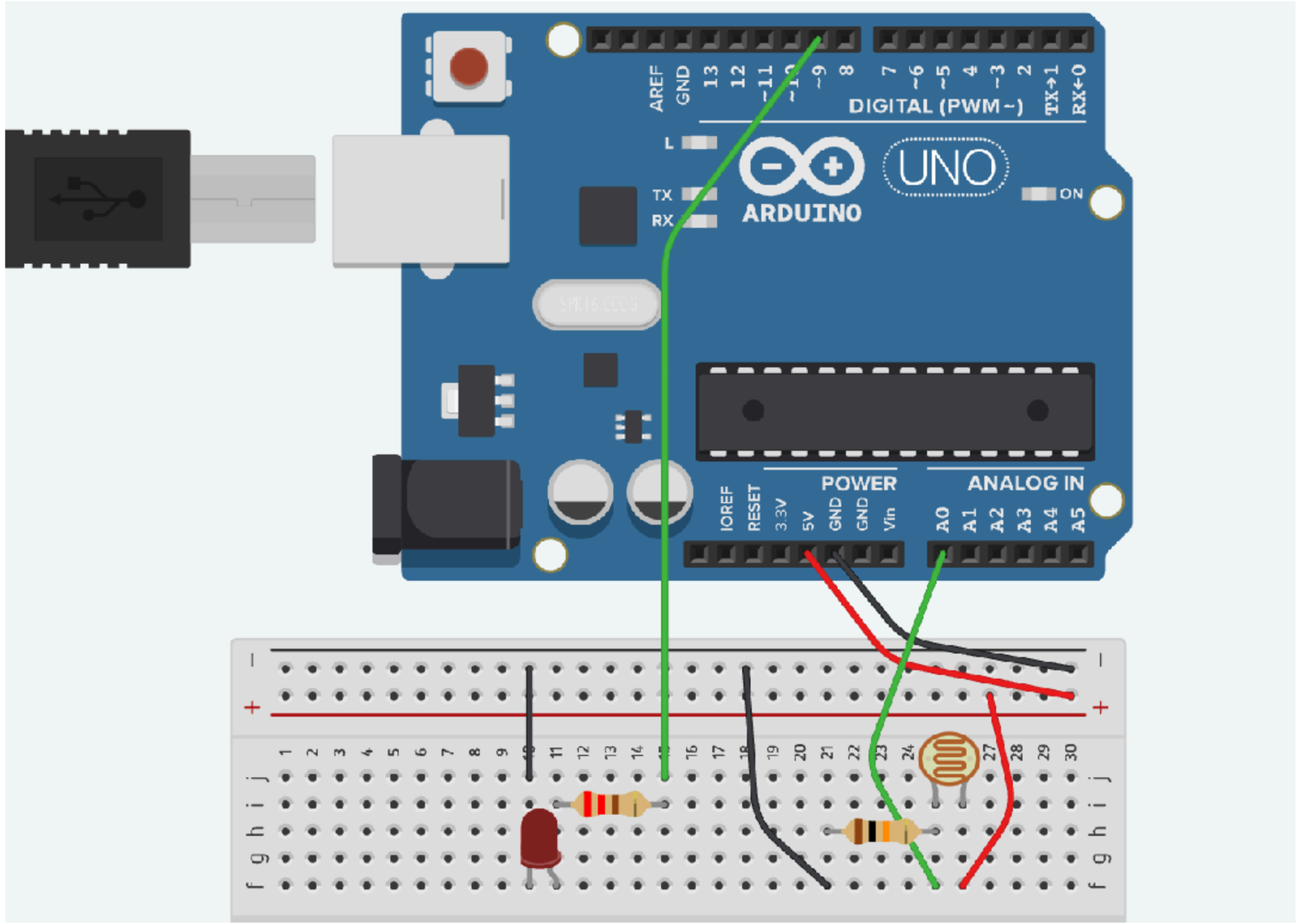
Only do these if you have already completed the examples on the last slide.

- Examples>Digital>BlinkWithoutDelay
- Examples>Digital>Debounce
- Examples>Digital>DigitalInputPullup
- Examples>Digital>StateChangeDetection

LDR
Light Dependent Resistor

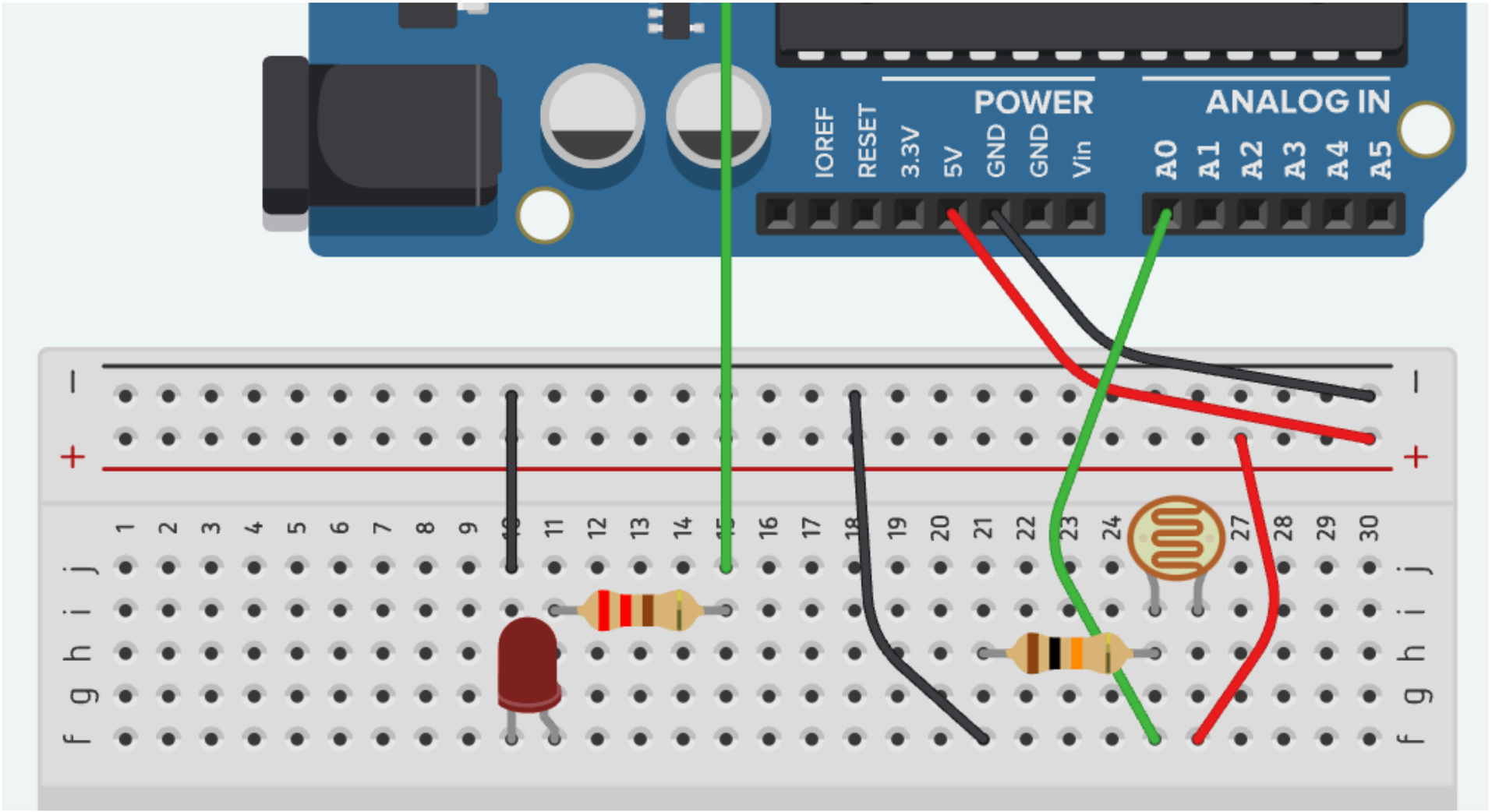
- Changes Resistance when exposed to light or dark.
- Must be used with another resistor (10K) to work.

Closeup

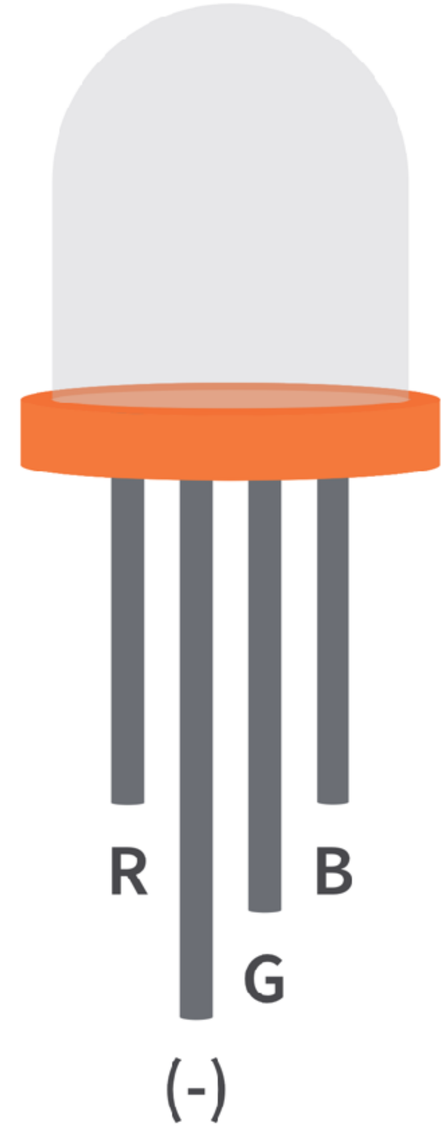

AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
ARDUINO
UNO
ON
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

Even Closer Up

/\ Up to Pin 9



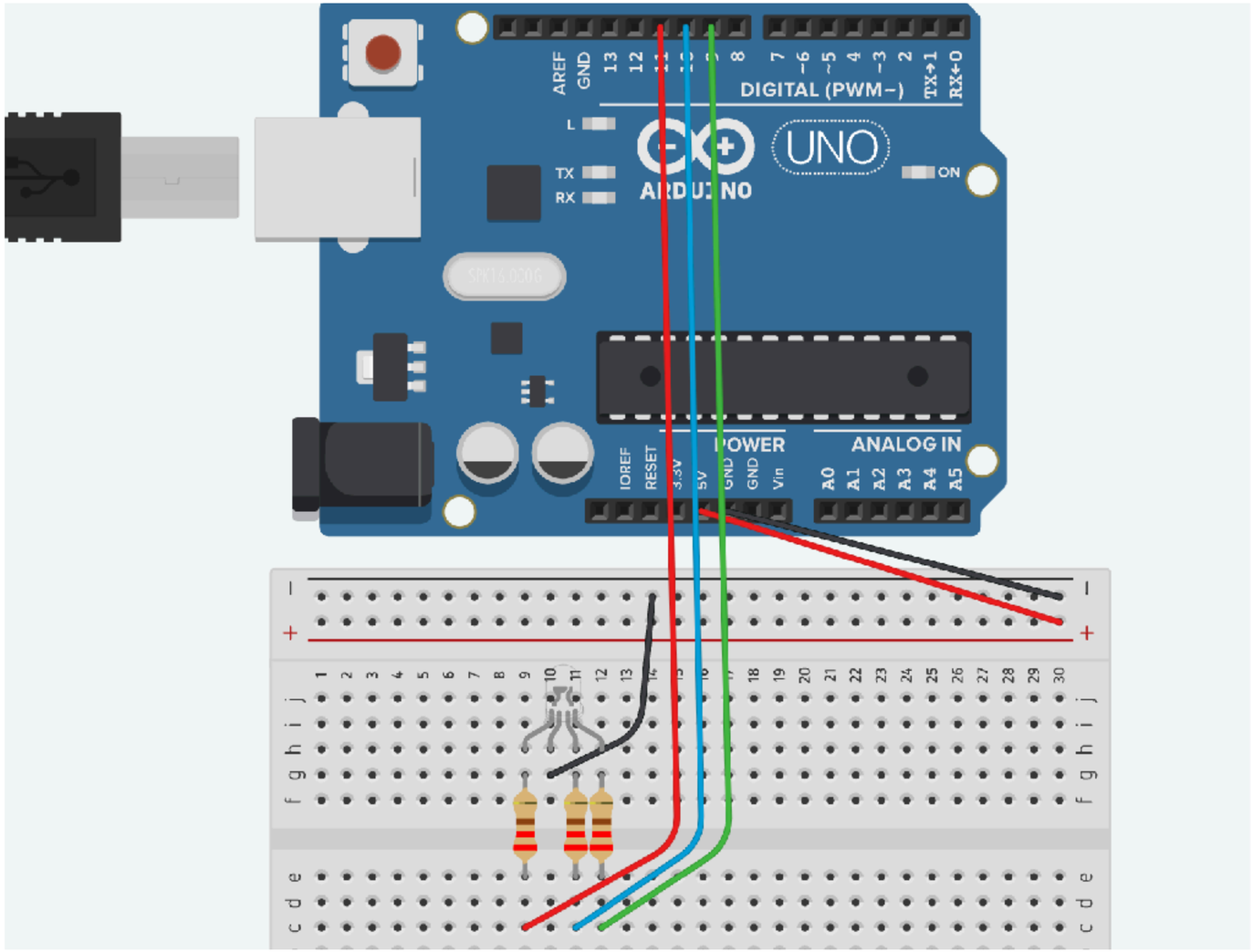
RGB LED



Common Cathode

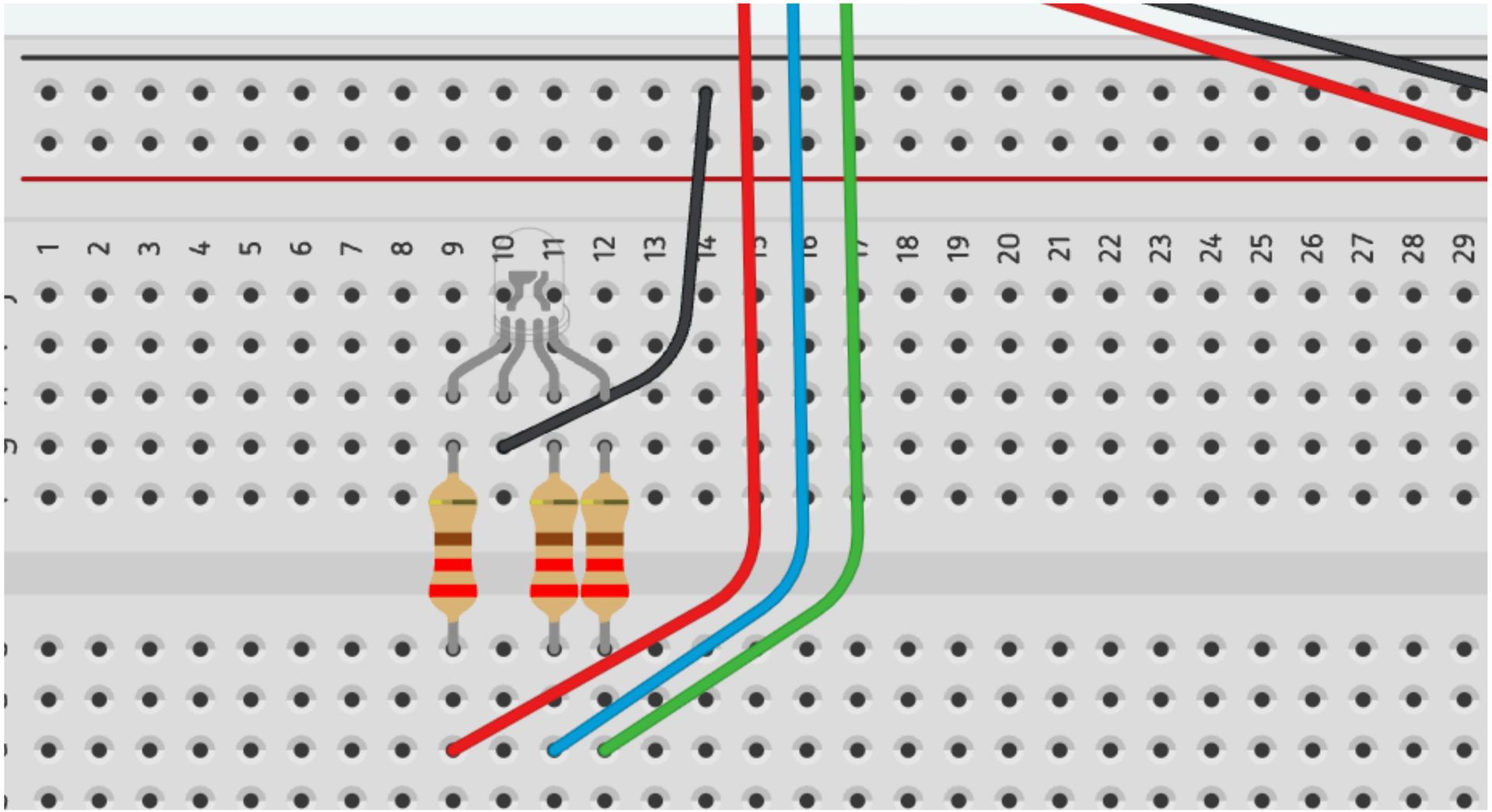
- Like 3 LEDs rolled into 1
- Can have Common Cathode or Common Anode.
- Common Cathode (Longest Leg) connects to Ground!

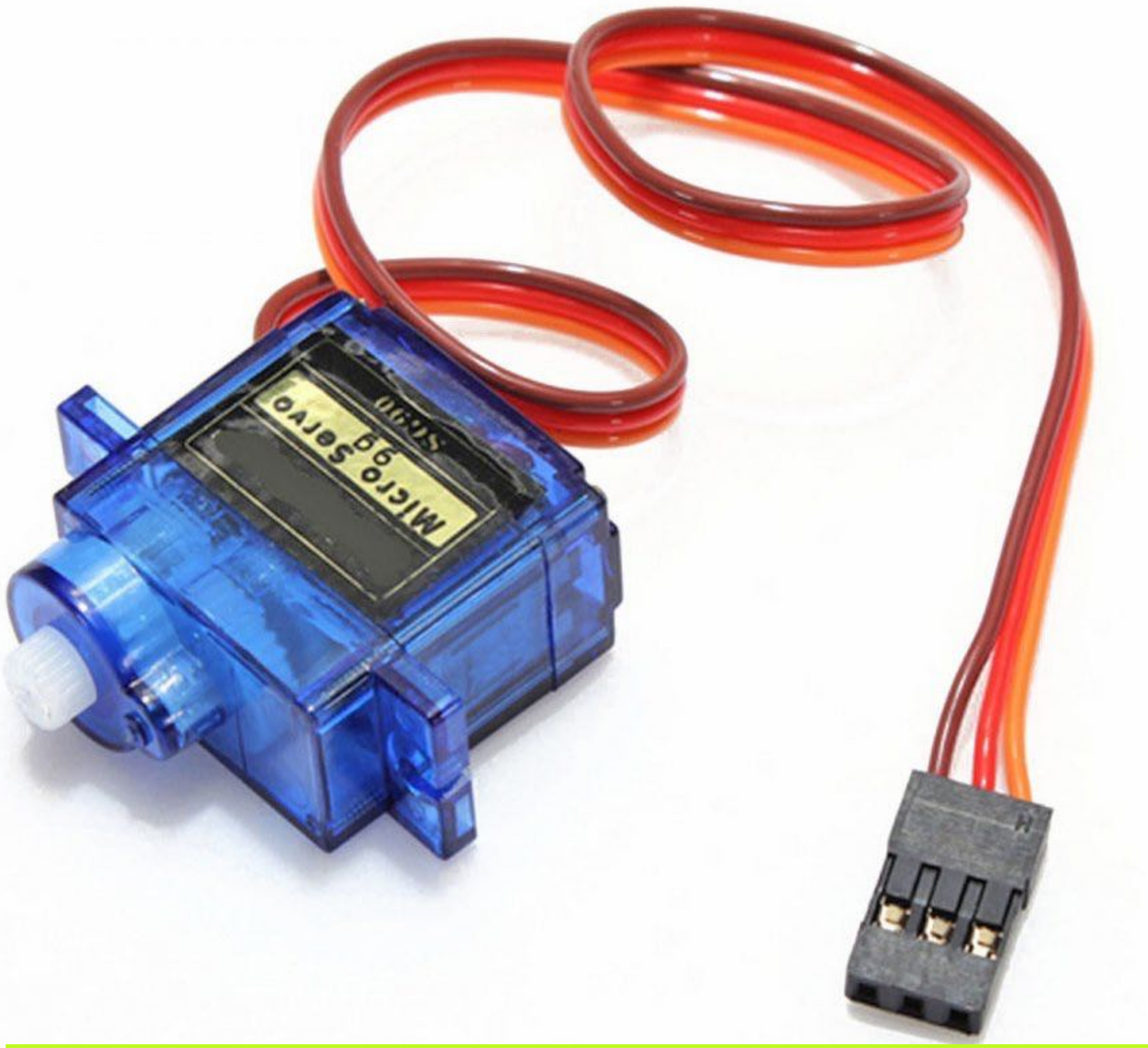
Closeup



Even Closer Up

/\ Up to Pins
9,10,11

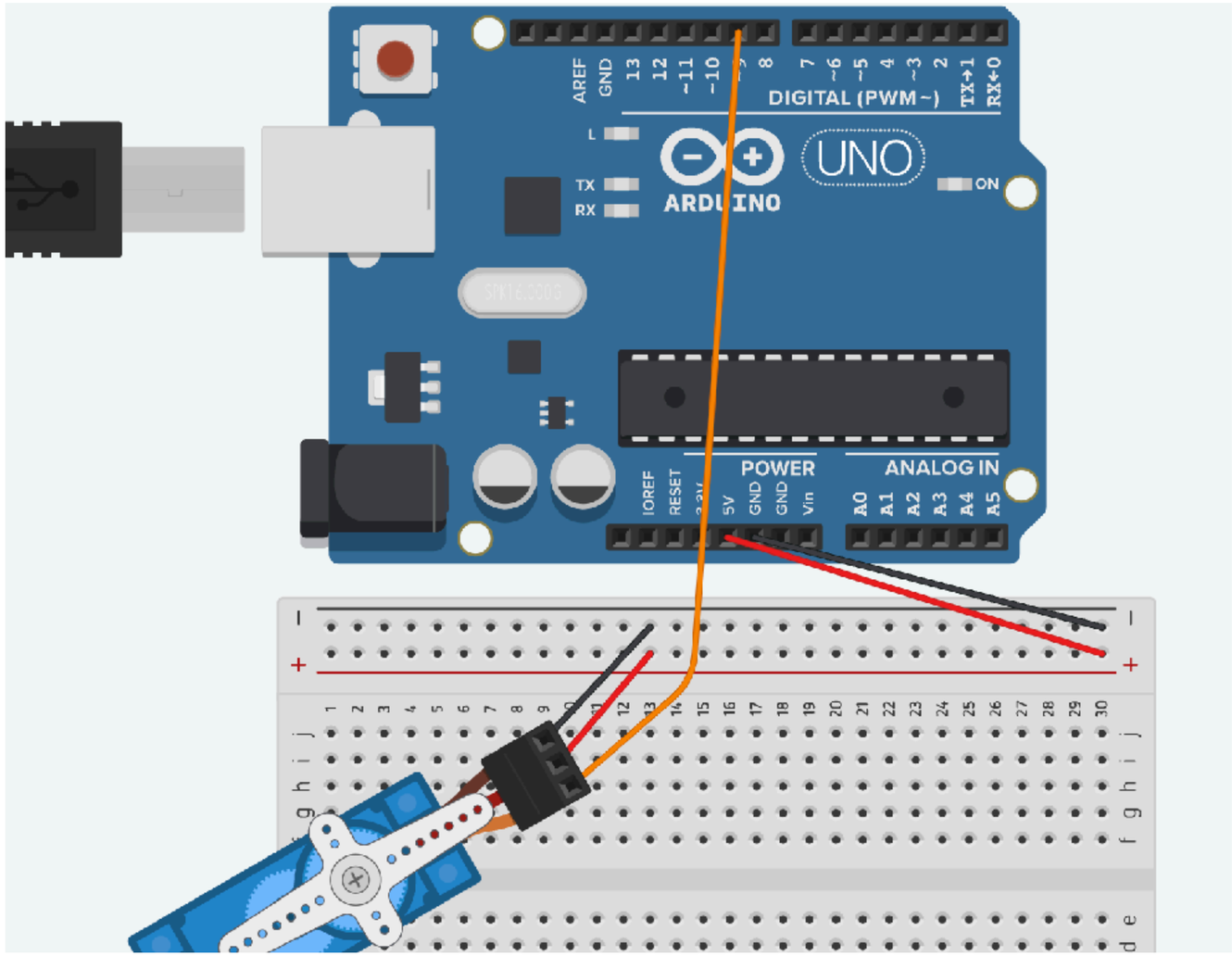




Servo Motor

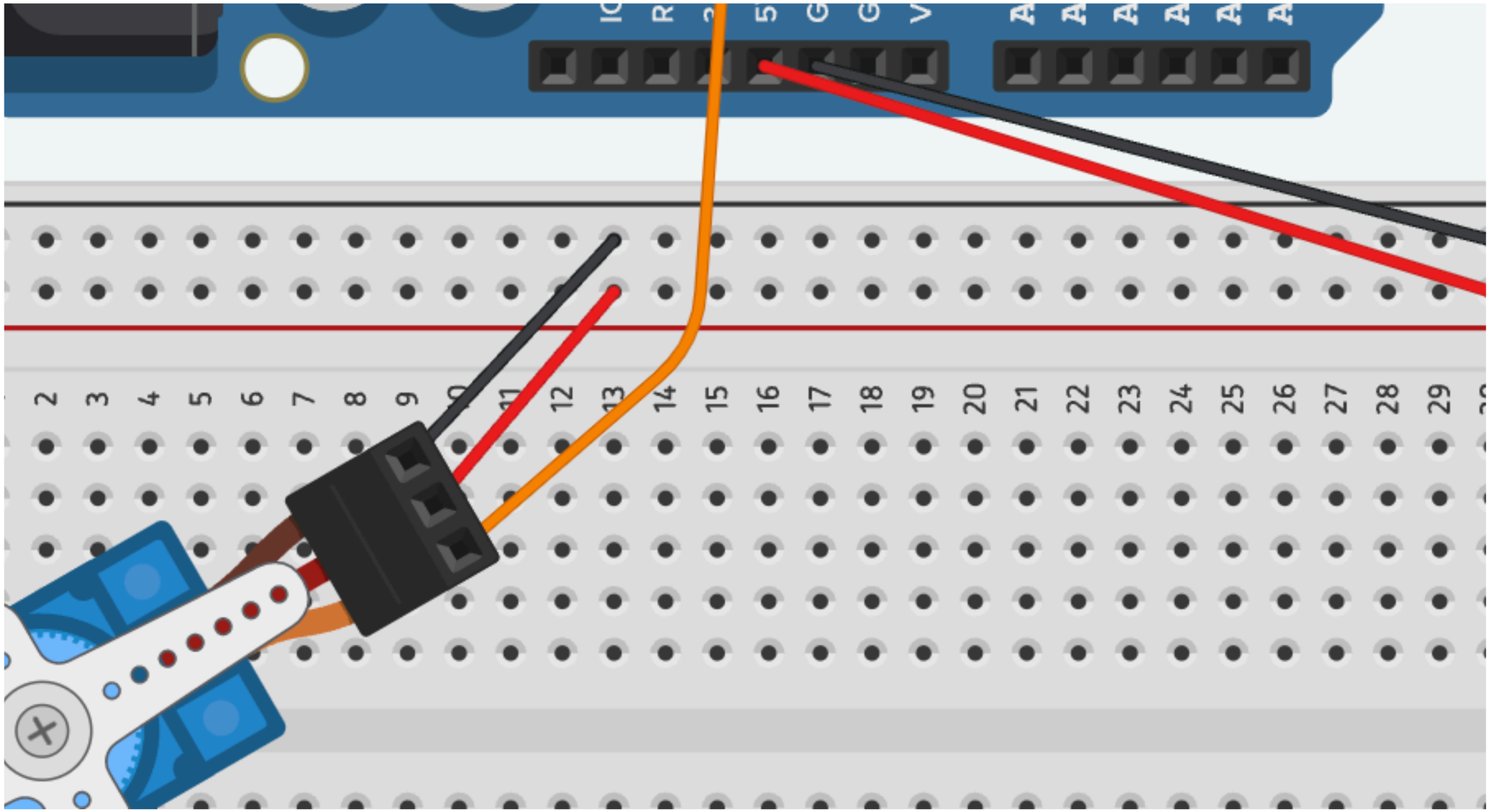
- Can move a small weight between 0 and 180 degrees
- Great for pressing buttons and switches or rotating small items
- Once you’re happy using this, there are other compatible motors that are much bigger and stronger but work exactly the same way

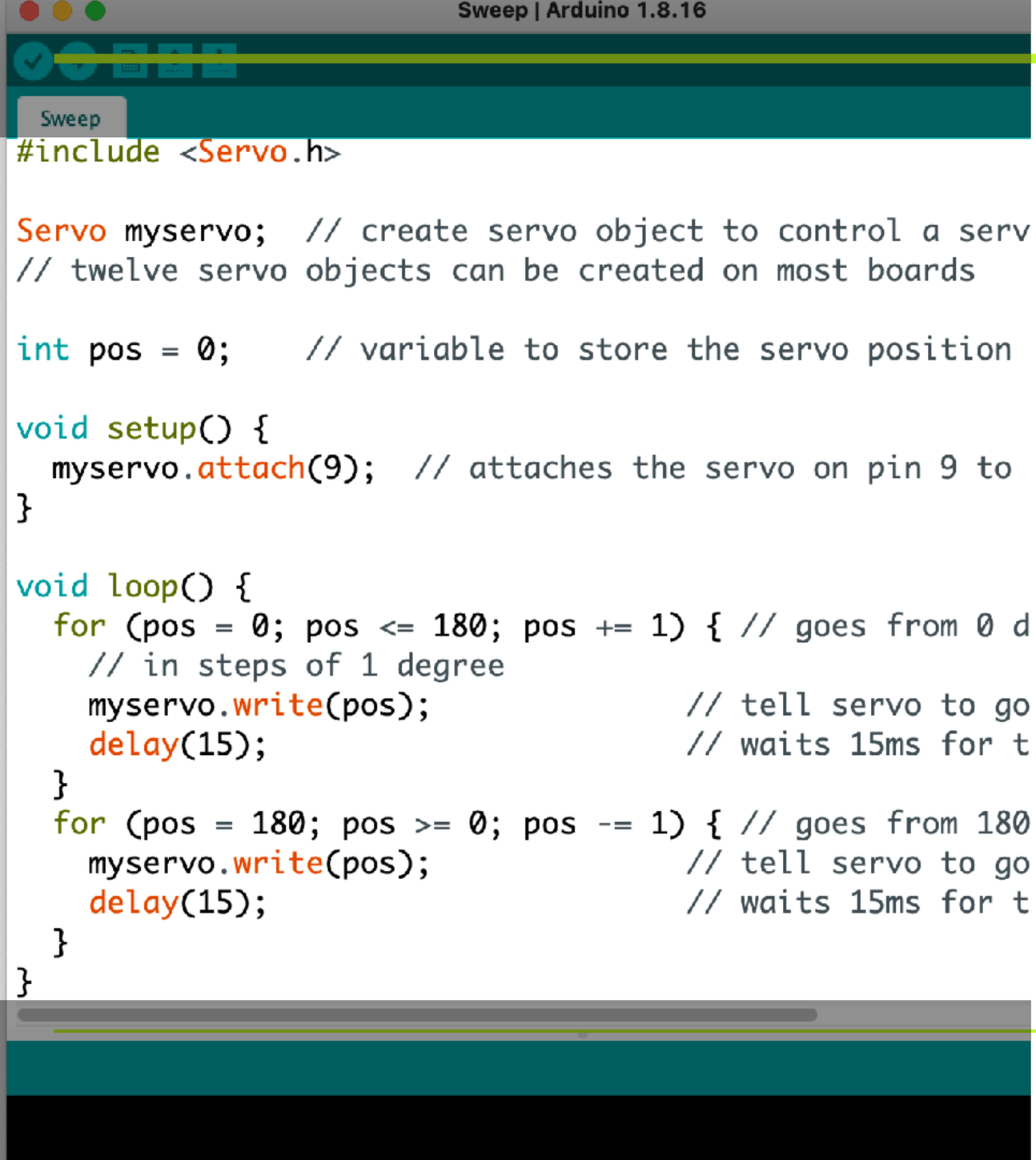
Closeup


AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
UNO
ON
POWER
ANALOG IN
IOREF
RESET
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

Even Closer Up

/\ Up to Pin 9

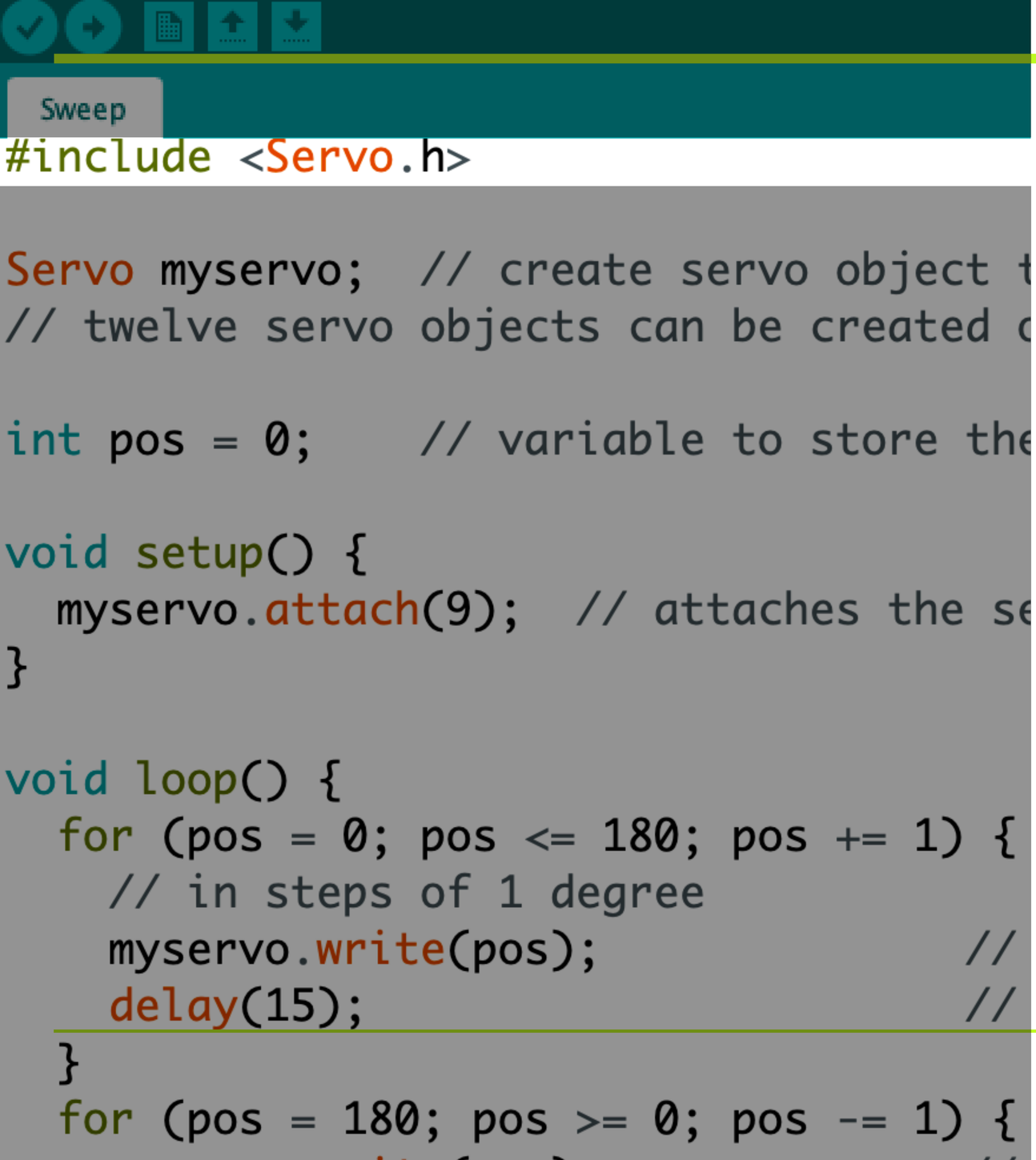




Servo Example

- Just like fading an LED we can move a Servo by writing different values to it.
- Because we are using an external library, the code looks a little different.

Servo Example



- First of all, because we are using a library, we need to include it
- We do this with this line which specifies which library we want to include

Sweep

```
#include <Servo.h>

Servo myservo;  // create servo o
// twelve servo objects can be cr

int pos = 0;    // variable to st

void setup() {
  myservo.attach(9);  // attaches
}

void loop() {
  for (pos = 0; pos <= 180; pos +
    // in steps of 1 degree
    myservo.write(pos);
```

Servo Example

- Next we need to make an instance of the servo class which contains the connection
- You can think of this like a variable that contains information on how to talk to a particular servo.

```
// twelve servo objects can be cr

int pos = 0;    // variable to st

void setup() {
  myservo.attach(9);  // attaches
}

void loop() {
  for (pos = 0; pos <= 180; pos +
    // in steps of 1 degree
    myservo.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos -
    myservo.write(pos);
    delay(15);
```

Servo Example

- Now we tell the servo instance which pin the servo is connected to, in this case pin 9

```
void setup() {
  myservo.attach(9);  // attaches
}

void loop() {
  for (pos = 0; pos <= 180; pos +=
    // in steps of 1 degree
    myservo.write(pos);
    delay(15);
  }
  for (pos = 180; pos >= 0; pos -=
    myservo.write(pos);
    delay(15);
  }
}
```

Servo Example

- Finally, use .write() to send the position to the servo

Libraries

The Arduino environment can be extended through the use of libraries, just like most programming platfc
Libraries provide extra functionality for use in sketches, e.g. working with hardware or manipulating data.
library in a sketch, select it from **Sketch > Import Library**.
A number of libraries come installed with the IDE, but you can also download or create your own. See the
instructions for details on installing libraries. There's also a tutorial on writing your own libraries. See the
Guide for information on making a good Arduino-style API for your library.

- Communication (754)
- Data Processing (164)
- Data Storage (94)
- Device Control (551)
- Display (319)
- Other (287)
- Sensors (664)
- Signal Input/Output (262)
- Timing (146)
- Uncategorized (128)

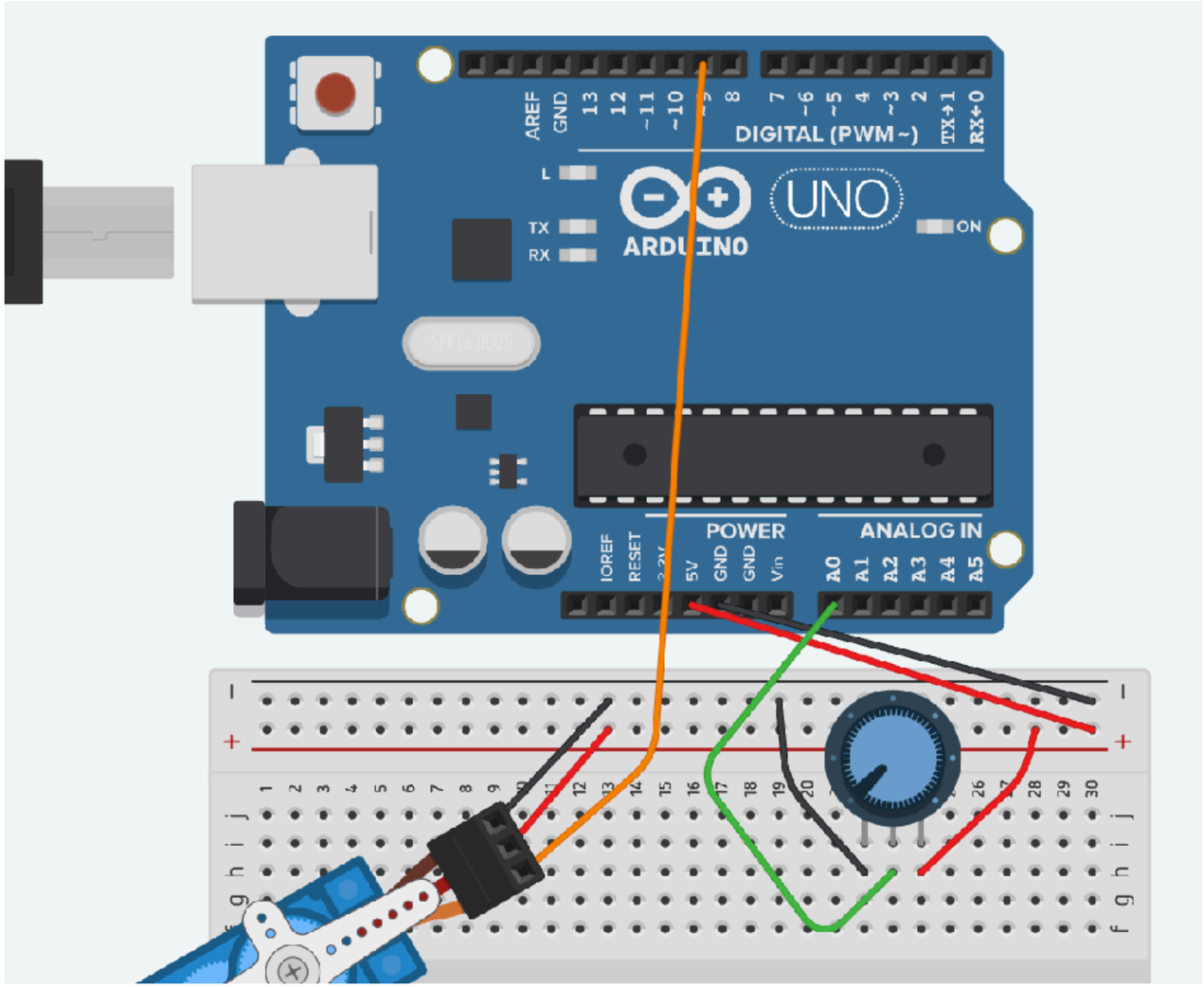
Standard Libraries

- EEPROM - reading and writing to "permanent" storage
- Ethernet - for connecting to the internet using the Arduino Ethernet Shield, Arduino Ethernet Shield 2 anc
  Arduino Leonardo ETH
- Firmata - for communicating with applications on the computer using a standard serial protocol.
- GSM - for connecting to a GSM/GRPS network with the GSM shield.
- LiquidCrystal - for controlling liquid crystal displays (LCDs)
- SD - for reading and writing SD cards
- Servo - for controlling servo motors
- SPI - for communicating with devices using the Serial Peripheral Interface (SPI) Bus
- SoftwareSerial - for serial communication on any digital pins. Version 1.0 and later of Arduino incorporate
  Hart's NewSoftSerial library as SoftwareSerial.
- Stepper - for controlling stepper motors
- TFT - for drawing text , images, and shapes on the Arduino TFT screen
- WiFi - for connecting to the internet using the Arduino WiFi shield

Servo Example

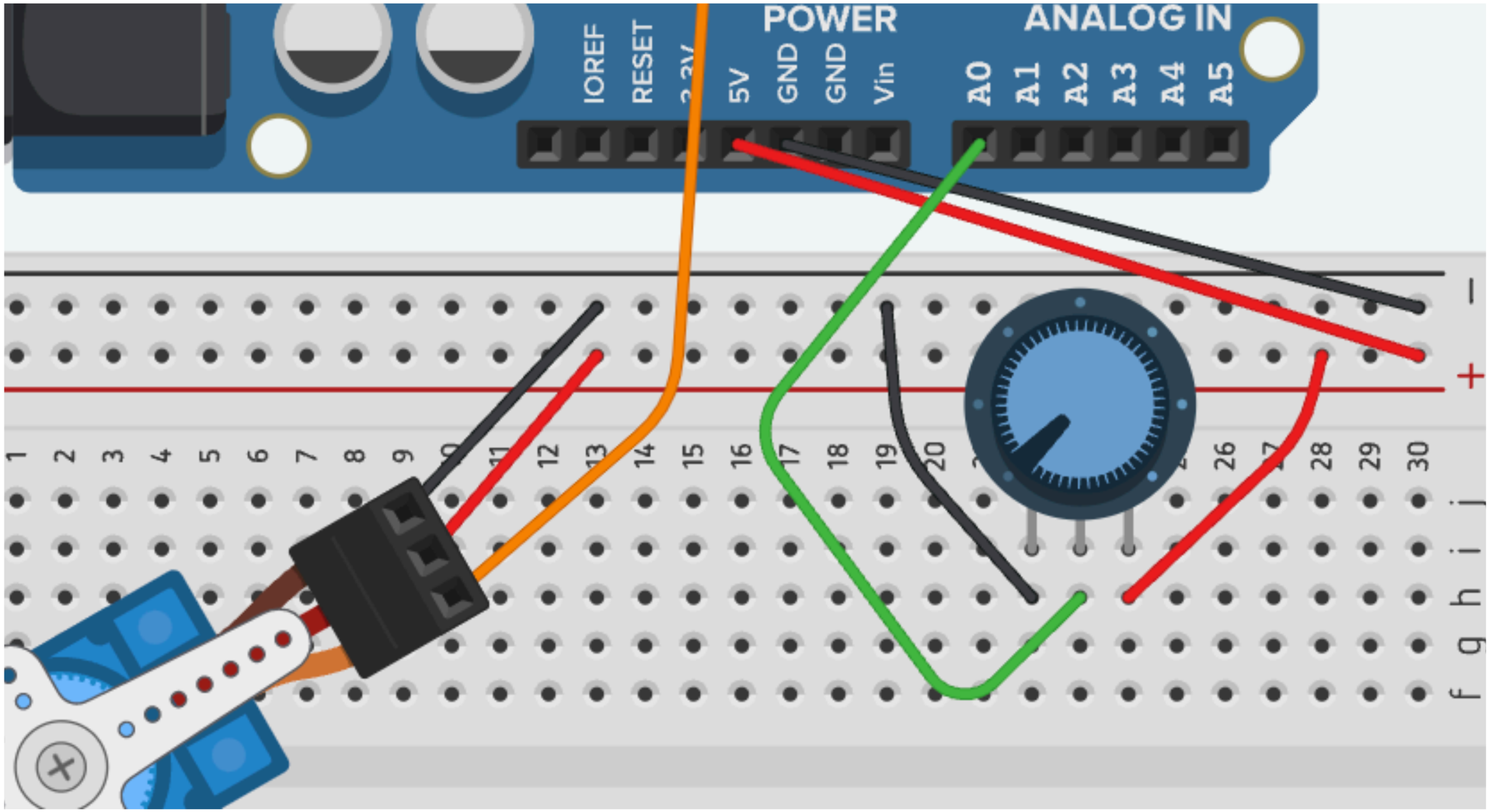
- The Arduino website has plenty of options for other libraries to add other functionality to your arduino.

Closeup


AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
L
TX
RX
ARDUINO
UNO
ON
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

Even Closer Up

/\ Up to Pin 9



Practical Exercises

Try out the following examples (the circuits to set up are described in the code) and work out what the code is doing….

- Examples>Servo>Sweep
- Examples>Servo>Knob
- RGB LED
  - There’s no example code for the RGB LEDs so you’ll have to make your own!
  - Control RGB LED with no inputs
  - Control RGB LED with potentiometer
  - Control RGB LED with LDR

Extension Exercises

Only do these if you have already completed the examples on the last slide. You might need to combine some code to make these work.

- Control the Servo with an LDR
- Control the Servo with 2 LDRs
- Control the Servo with a potentiometer and an LDR
- Build a musical light and movement Theremin:
  - Using a buzzer, servo and LED

Extension Exercises

If you get all that done!

Extension 1

- Build a 2 player game using a separate input for each player
- The game should be competitive and have a goal and a clear winner
  - Reimagine the game to make it collaborative so the players have to work together to win.

Extension 2

- Build a Digital Hourglass but instead of sand use lights
  - To time 30 seconds
  - Like an hour glass timer that starts when you turn it over and lights an LED every 5 seconds

Some Useful Things

- Blink Without Delay
- Digital Input Pullup
- State Change Detection

Second Assignment

Expect the Unexpected!

You should make a physical installation that subverts my expectations.

The goal of this mini-project is to build a physical installation (for example this could be an art work, game, some thing useful, a product, an interactive installation etc) that has at least one input and one output and does something unexpected. Think about what the user/viewer will: SEE, HEAR, FEEL. Think about how to prototype your installation and test it, how will people know what to do, what feedback will they get, will there be interaction. Think about what you would expect your installation to do and then make it do the opposite.

Please include, at the minimum, the following: - one input (sensor) - one output (actuator)

Upload this assignment to the VLE and include: A short video of the project (1 minute or less) , your code, a schematic of your circuit. The schematic can be hand drawn or created using other software you choose - for example Fritzing or EasyEDA. ALSO Post the short video of your device to the class forum so we can all see everyone's work. Please upload everything to the VLE and forum.

Please think of this brief as an opportunity to try things out, the result does not have to be perfect, feel free to try things and fail, to play, to experiment, to get things wrong.