technocamps

UNDEB EWROPEAIDD
EUROPEAN UNION

Llywodraeth Cymru
Welsh Government

Cronfa Gymdeithasol Ewrop
European Social Fund

Prifysgol Abertawe
Swansea University

CARDIFF UNIVERSITY
PRIFYSGOL CAERDYDD

PRIFYSGOL BANGOR UNIVERSITY

Cardiff Metropolitan University
Prifysgol Metropolitan Caerdydd

i.t.wales

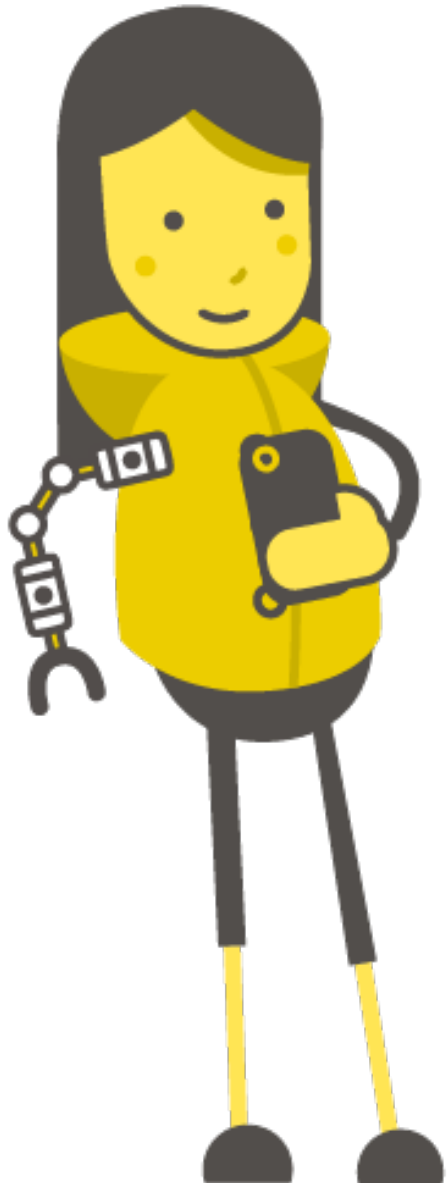
PRIFYSGOL ABERYSTWYTH UNIVERSITY

PRIFYSGOL glyndŵr Wrecsam
Wrexham glyndŵr UNIVERSITY

University of South Wales
Prifysgol De Cymru

technocamps

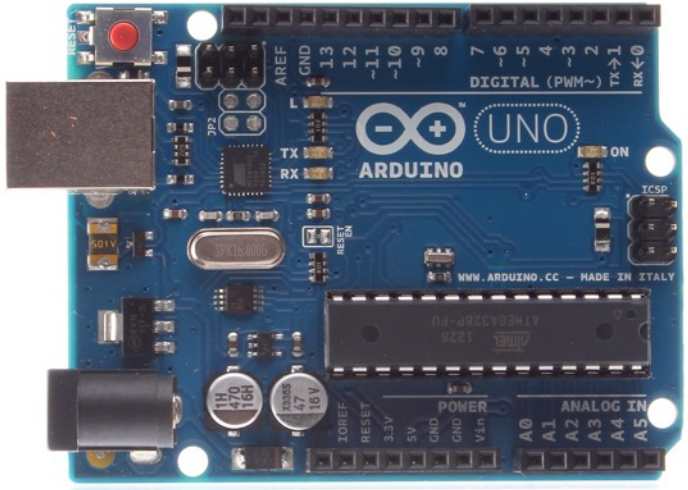
Arduino



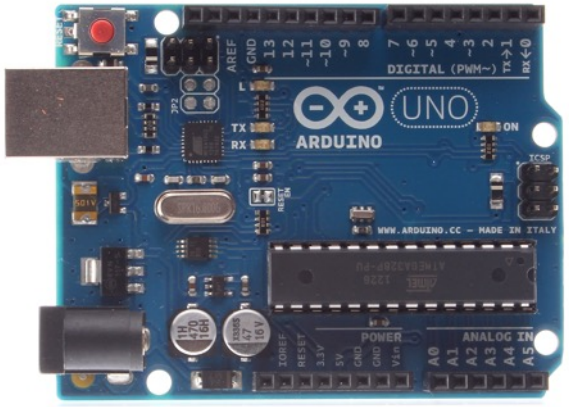
Arduino

Open-source electronic prototyping platform enabling users to create interactive electronic objects.

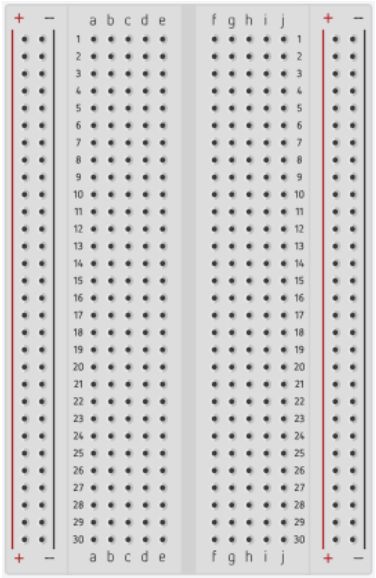
We will be making both the electronic circuit and writing the code to make it work!



Circuit Components

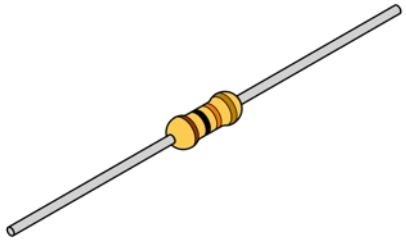


Arduino

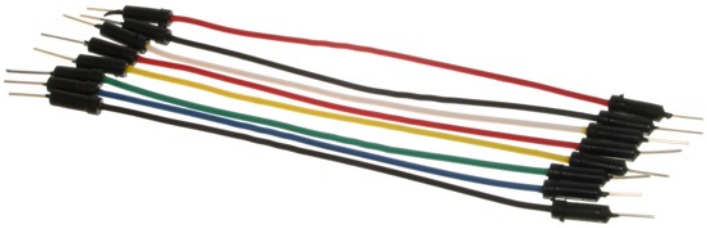


Breadboard

LED



Resistors

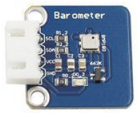


Jumper Wires

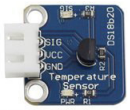
LDR (Light Sensor)

technocamps

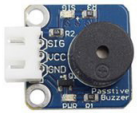
Circuit Components: Sensors



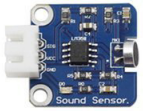
Barometer



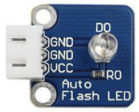
DS18B20 Temperature



Passive buzzer



Sound Sensor



Auto-flash LED

Dual-color LED

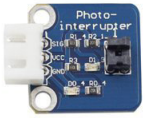
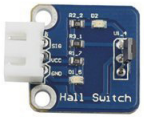
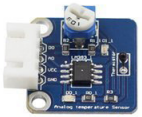


Photo-interrupter

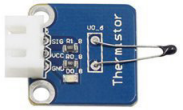


Switch Hall

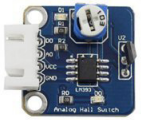

Analog temperature

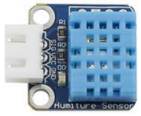
Flame Sensor

Photoresistor

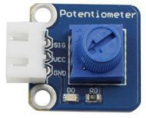


Thermistor module

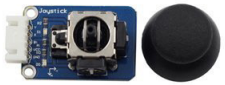

Analog Hall



Humiture sensor

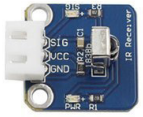


Potentiometer

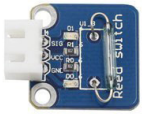


Joystick PS2 module

Active buzzer

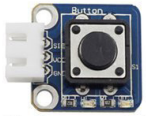


Infrared-Receiver



Reed Switch

MQ-2 Gas Sensor



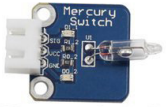
Button module

Laser Transmitter

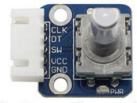
RGB LED

Relay Module


ADDA Converter

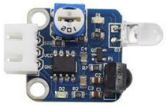


Mercury Switch


Rotary Encoder

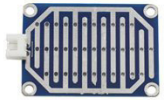

Tracking sensor

Color Sensor

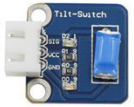

Obstacle Avoidance

RTC-DS1302

Ultrasonic



Raindrop Sensor


1602LCD

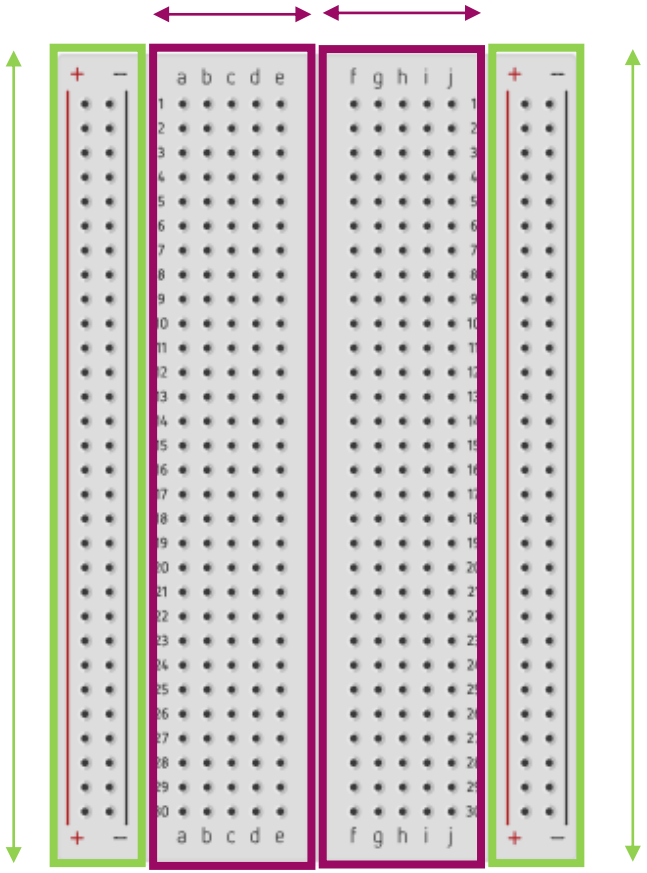


Tilt Switch

Touch Switch

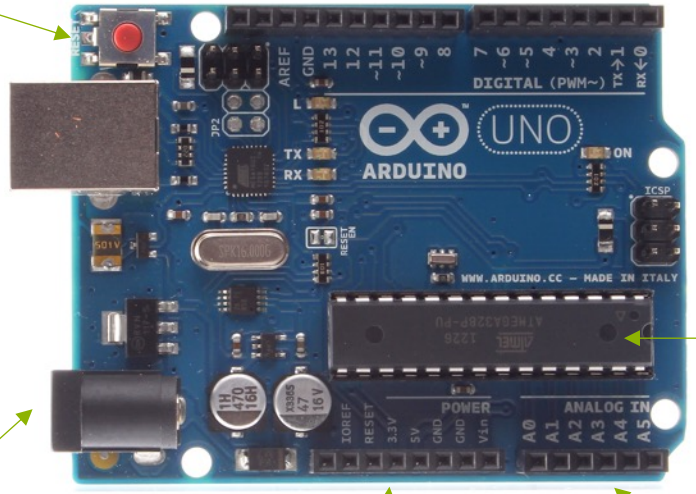

Remote Controller

Circuit Components: Breadboard



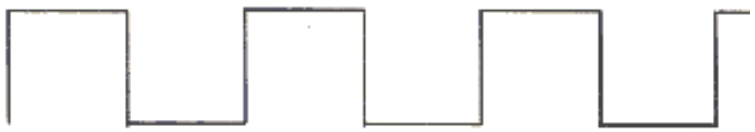
technocamps

Arduino in More Detail



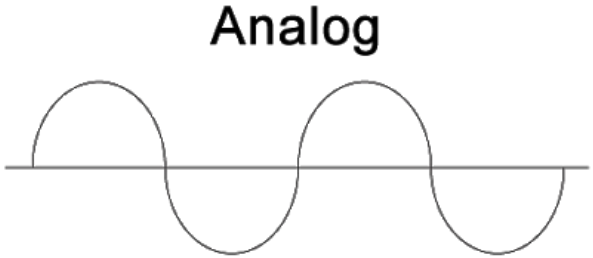
Digital vs Analogue

Digital signals are either on or off, either 0 or 1



Digital Signal

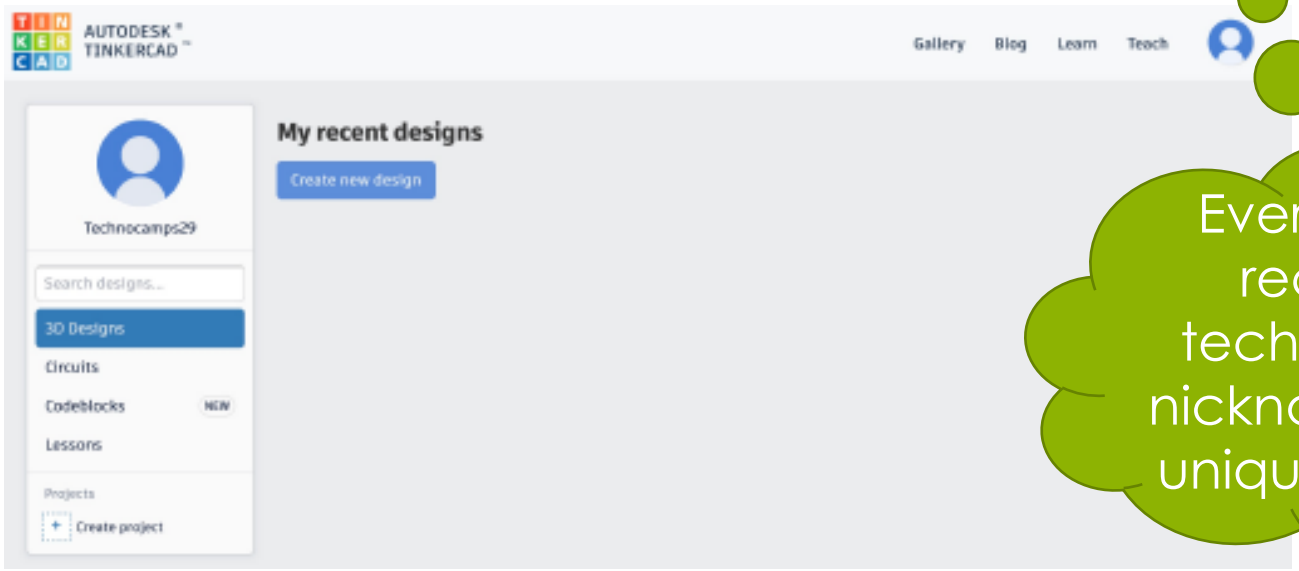
Analogue signals cover a large range of values. For us today analogue signals will range between 0-255 or 0-1023



What about analogue output?

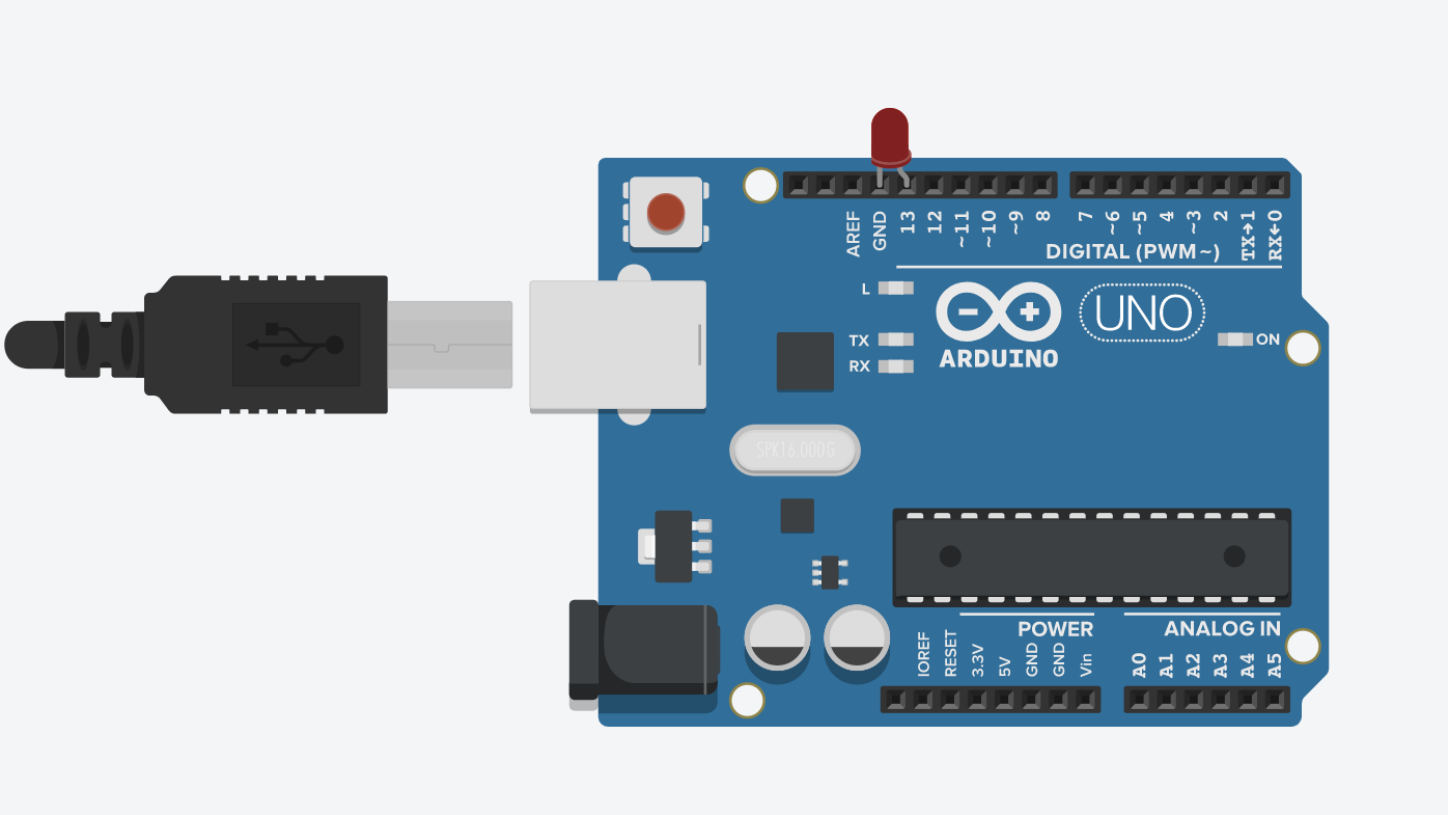
Tinkercad

1. Go to www.tinkercad.com/joinclass
2. Enter the class code
3. For your nickname enter technocamps1
4. Click on “circuits” and then “create new circuit”



Everyone will receive a technocamps nickname with a unique number!

Blinking LED



Blinking LED

```
// Example 01 : Blinking LED

const int LED = 13; // LED connected to digital pin 13

void setup()
{

  pinMode(LED, OUTPUT);   // sets the digital pin as output

}

void loop()
{

  digitalWrite(LED, HIGH);  // turn the LED on
  delay(1000);              // waits for a second
  digitalWrite(LED, LOW);   // turns the LED off
  delay(1000);              // waits for a second

}
```

Setup and Loop

setup () {

this is where you write all the code that you want to execute once at the beginning of your program

}

loop () {

this contains the core of your program which is executed over and over again

}

// ← this is how we write comments. Comments are good and used to let other people read and understand your code more easily

The Code – Step by Step

```
// Example 01 : Blinking LED
```

This is a useful comment which gives an indication of what the program does

```
const int LED = 13; // LED connected to digital pin 13
```

const int means that the variable LED is an integer number that cannot be changed. We have set the variable LED to the value 13.

Every time the Arduino "sees" the word LED it internally replaces it with the value 13.

The Code – Step by Step

```
void setup()
{

  pinMode(LED, OUTPUT);   // sets the digital pin as output

}
```

Everything inside the curly brackets { } is part of the setup() function.

pinMode(LED, OUTPUT) tells the Arduino that the pin called LED i.e. pin number 13 is set up so that it outputs something rather than is used for input.

The Code – Step by Step

```
void loop()
{

  digitalWrite(LED, HIGH);  // turn the LED on
  delay(1000);              // waits for a second
  digitalWrite(LED, LOW);   // turns the LED off
  delay(1000);              // waits for a second

}
```

Everything inside the curly brackets is part of the loop() function.

digitalWrite(LED, HIGH) sets the value of pin 13 to be HIGH i.e. on

delay(1000) delays the program for 1000 milliseconds i.e. 1 second

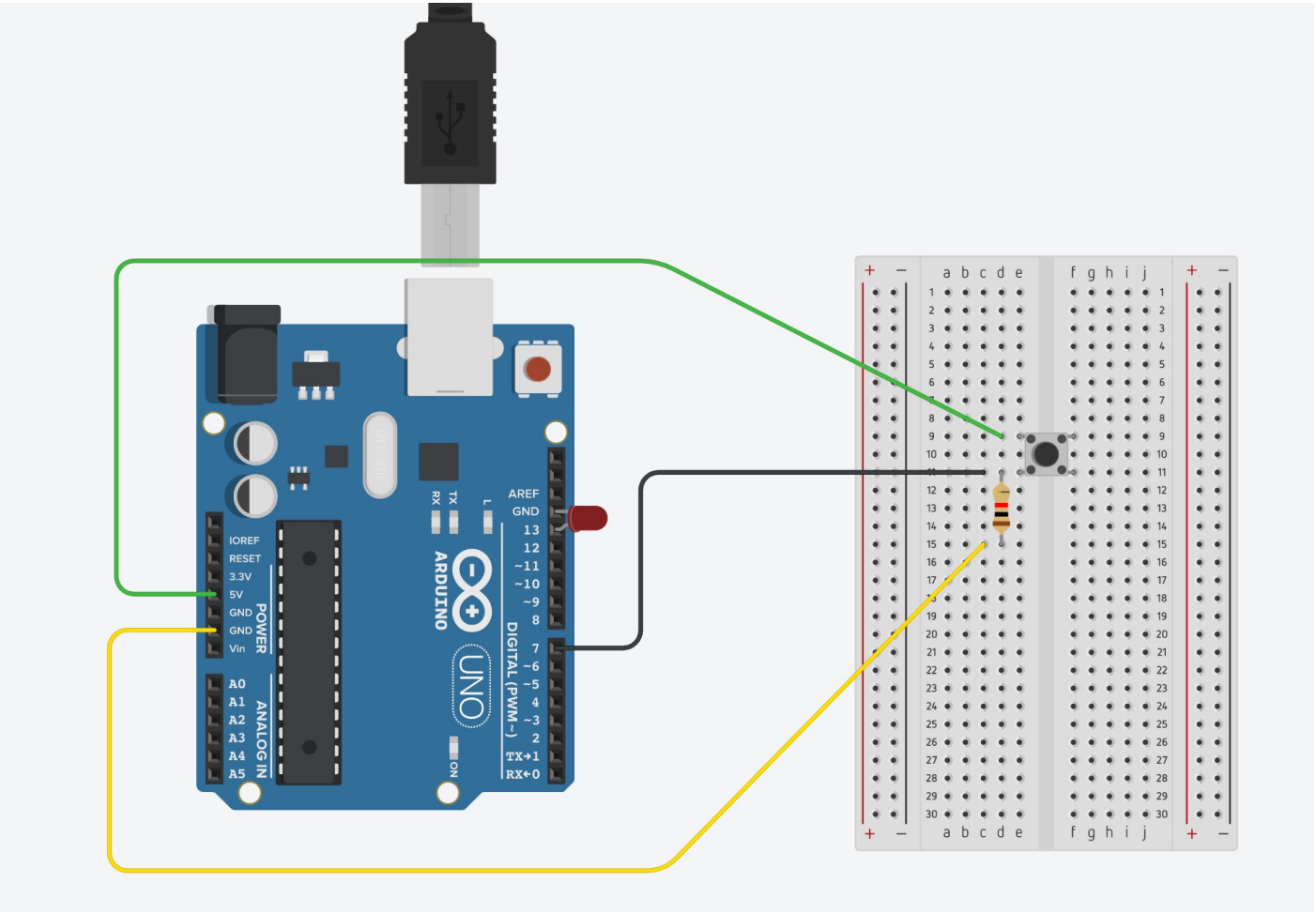
digitalWrite(LED, LOW) sets the value of pin 13 to be LOW i.e. off

Full Code

- Turns pin 13 into an output
- Enters a loop
- Switches on the LED connected to pin 13
- Waits 1 second
- Switches off the LED connected to pin 13
- Waits 1 second
- Goes back to the beginning of the loop

Lets add a button

LED with a Button



technocamps

Button LED

```
// Example 02: Turn on LED while the button is prssed

const int LED = 13;      // the pin for the LED
const int BUTTON = 7;    // the input pin where the pushbutton is connected
int val = 0;             // val will be used to store the state of the input pin


void setup()
{

  pinMode(LED,OUTPUT);     // tell Arduino LED is an output
  pinMode(BUTTON,INPUT);   // tell Arduino BUTTON is an input

}
```

Button LED

```
void loop()
{

  val = digitalRead(BUTTON);  // read input value and store it

  // check whether the input is HIGH (button pressed)

  if (val==HIGH)
  {
    digitalWrite(LED, HIGH);  // turn LED ON
  }
  else
  {
    digitalWrite(LED, LOW);
  }

}
```

'If' Statement

If you're wearing a jumper stand up

– ***'if' statement***

If you have blue eyes clap your hands, if you have any other colour eyes wave your hands

– ***'if, else' statement***

So what does the code do?

Step by Step

```
'
// Example 02: Turn on LED while the button is prssed

const int LED = 13;      // the pin for the LED
const int BUTTON = 7;    // the input pin where the pushbutton is connected
int val = 0;             // val will be used to store the state of the input pin
```

const int LED = 13 ← the variable LED is a constant integer number permanently set to 13

const int BUTTON = 7 ← the variable BUTTON is a constant integer number permanently set to 7

int val = 0 ← the variable val is an integer currently set to 0

Step by Step

```
void setup()
{

  pinMode(LED,OUTPUT);     // tell Arduino LED is an output
  pinMode(BUTTON,INPUT);   // tell Arduino BUTTON is an input

}
```

pinMode(LED, OUTPUT) ← sets pin 13 to an output pin

pinMode(BUTTON, INPUT) ← sets pin 7 to an input pin

Step by Step

```
void loop()
{

  val = digitalRead(BUTTON);  // read input value and store it
```

Inside our loop() function we first set the variable val to the value read by the Arduino from the pin 7. As it is a digital pin it will either be 0 or 1, on or off, LOW or HIGH.

```
  if (val==HIGH)
  {
    digitalWrite(LED, HIGH);  // turn LED ON
  }
```

If the button is pressed i.e. the value is HIGH then digitalWrite pin 13 to HIGH i.e. send power to pin 13.

Step by Step

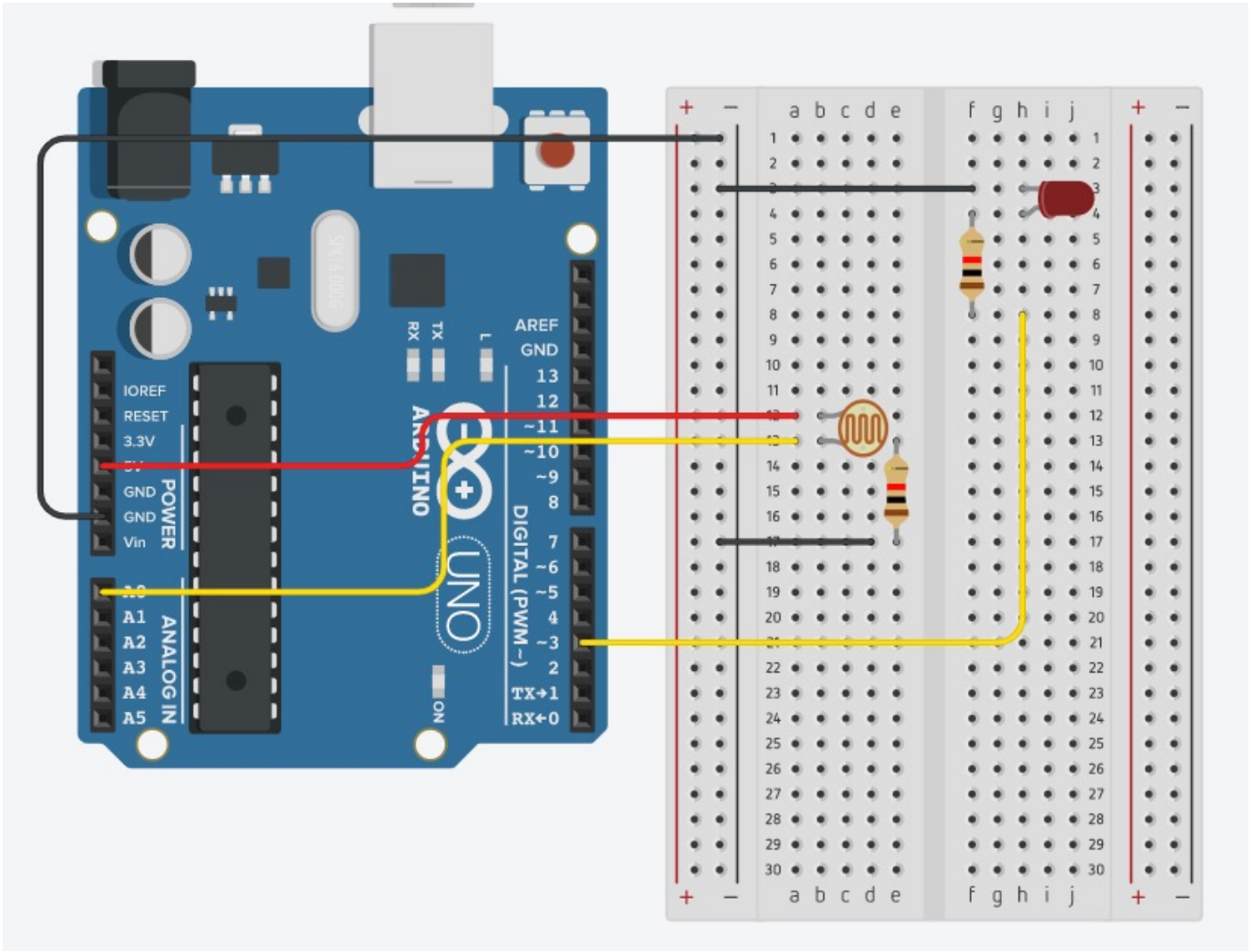
```
  else
  {
    digitalWrite(LED, LOW);
  }

}
```

If the button is not pressed and therefore val does not equal HIGH then set the pin 13 to LOW i.e. do not power pin number 13.

technocamps

Building Arduinos

Smart Light


IOREF
RESET
3.3V
GND
GND
Vin
POWER
A1
A2
A3
A4
A5
ANALOG IN
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
TX
RX
L
ON
UNO
a b c d e
f g h i j

Smart Light

```
// Example: Smart Light

int sensorPin = 0;    // Sensor connected to pin 0
int lightPin = 3;     // LED connected to pin 3

int threshold = 400; // An analog value to turn the LED on/off


void setup() {
  Serial.begin(9600);          // Reset the Arduine to communitcate via USB

  pinMode(lightPin,OUTPUT);  // Set the LED (pin 3) to Output
}
```

Smart Light

```
void loop() {
  int sensorValue = analogRead(sensorPin); // Read the value of the Light Sensor
  Serial.println(sensorValue,DEC);         // Send the light sensor value back to the computer


  if (sensorValue < threshold){            // If the sensor value is below the threshold
    digitalWrite(lightPin, HIGH);          // set the LED to HIGH
  }

  if (sensorValue > threshold){            // If the sensor value is above the threshold
    digitalWrite(lightPin, LOW);           // set the LED to LOW
  }
}
```

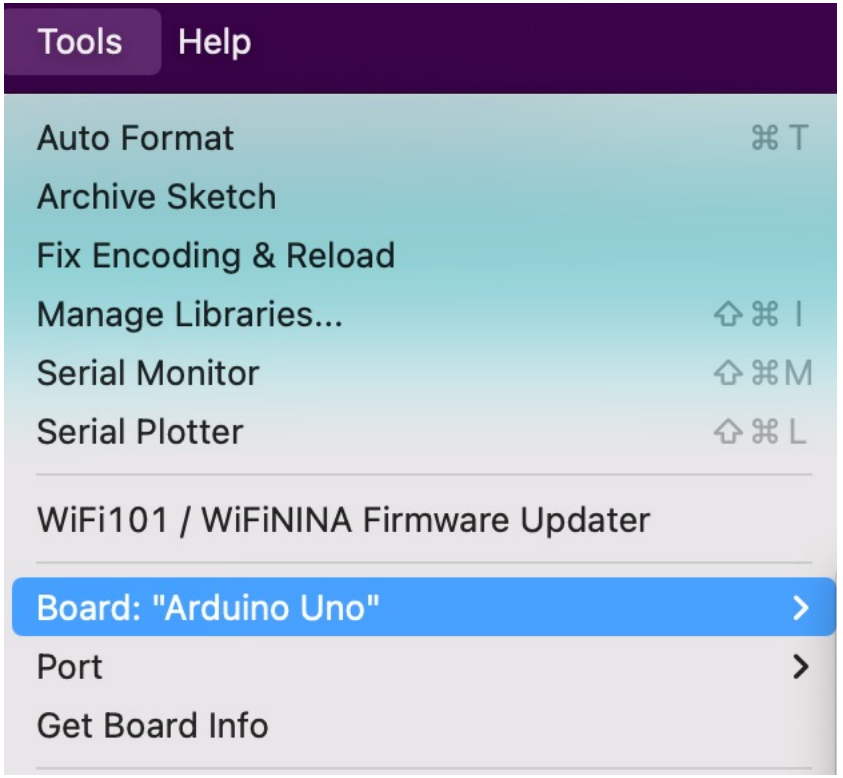
Choosing the Right Settings

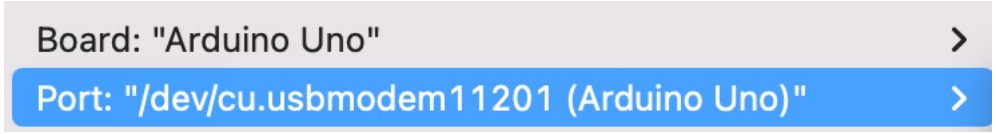
After connecting the Arduino via USB you'll have to ensure the correct settings are chosen to upload the code.

Under the "Tools" menu, set the following:

Board > Arduino Uno

Port > ...usbmodem...





Verifying and Uploading

Before uploading the code to the Arduino, you can verify it is correct by using the tick button.

If the code is verified successfully you can upload the code using the arrow button.

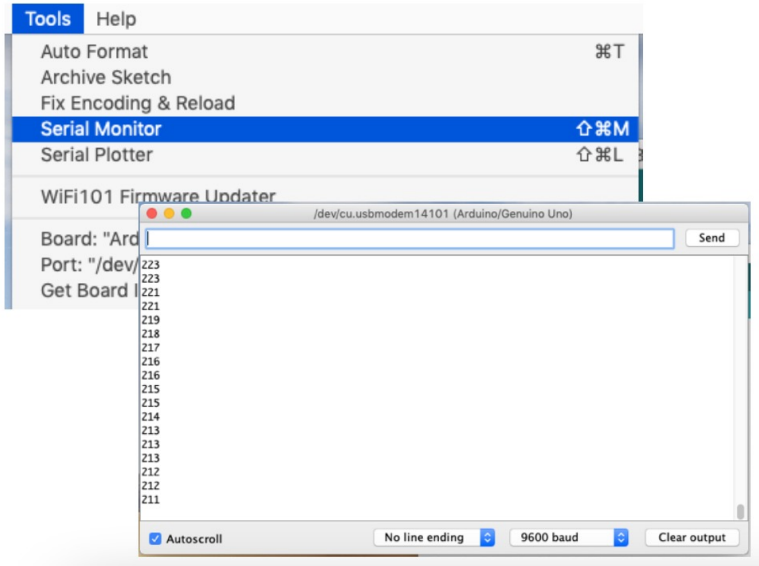
Remember, to be able to upload you must set the correct board and port in the tools menu!

Using the Serial Monitor

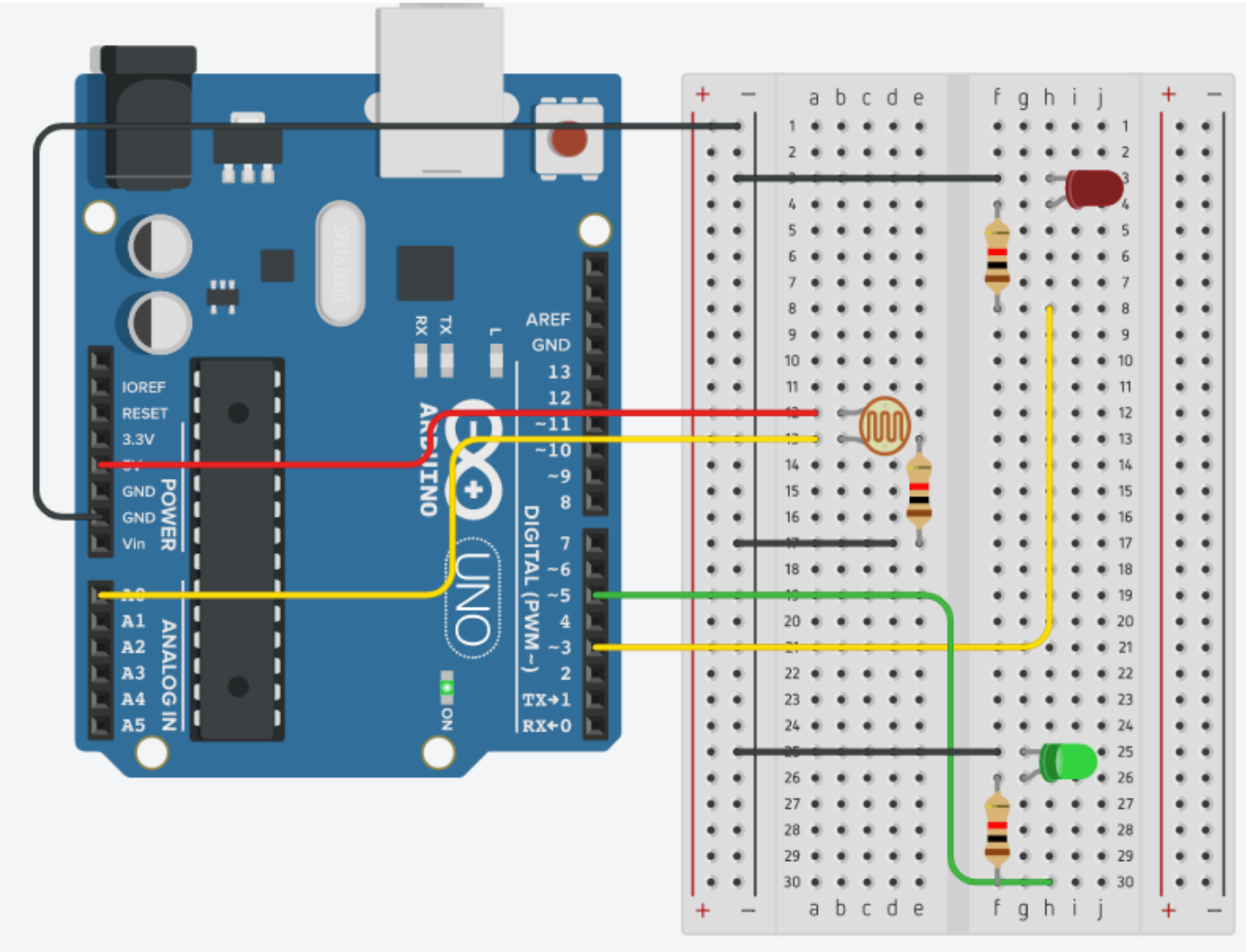
This section of code allows us to see the values from the light sensor printed on the computer screen.

```
Serial.begin(9600);          // Reset the Arduine to communitcate via USB

Serial.println(sensorValue,DEC);          // Send the light sensor value back to the computer
```

To open the serial monitor and see the values printed in real time, open it from the Tools menu:



Two LEDs


IOREF
RESET
3.3V
GND
GND
Vin
POWER
A1
A2
A3
A4
A5
ANALOG IN
TX
RX
L
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
ON
UNO
a b c d e
f g h i j
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30

Two LEDs

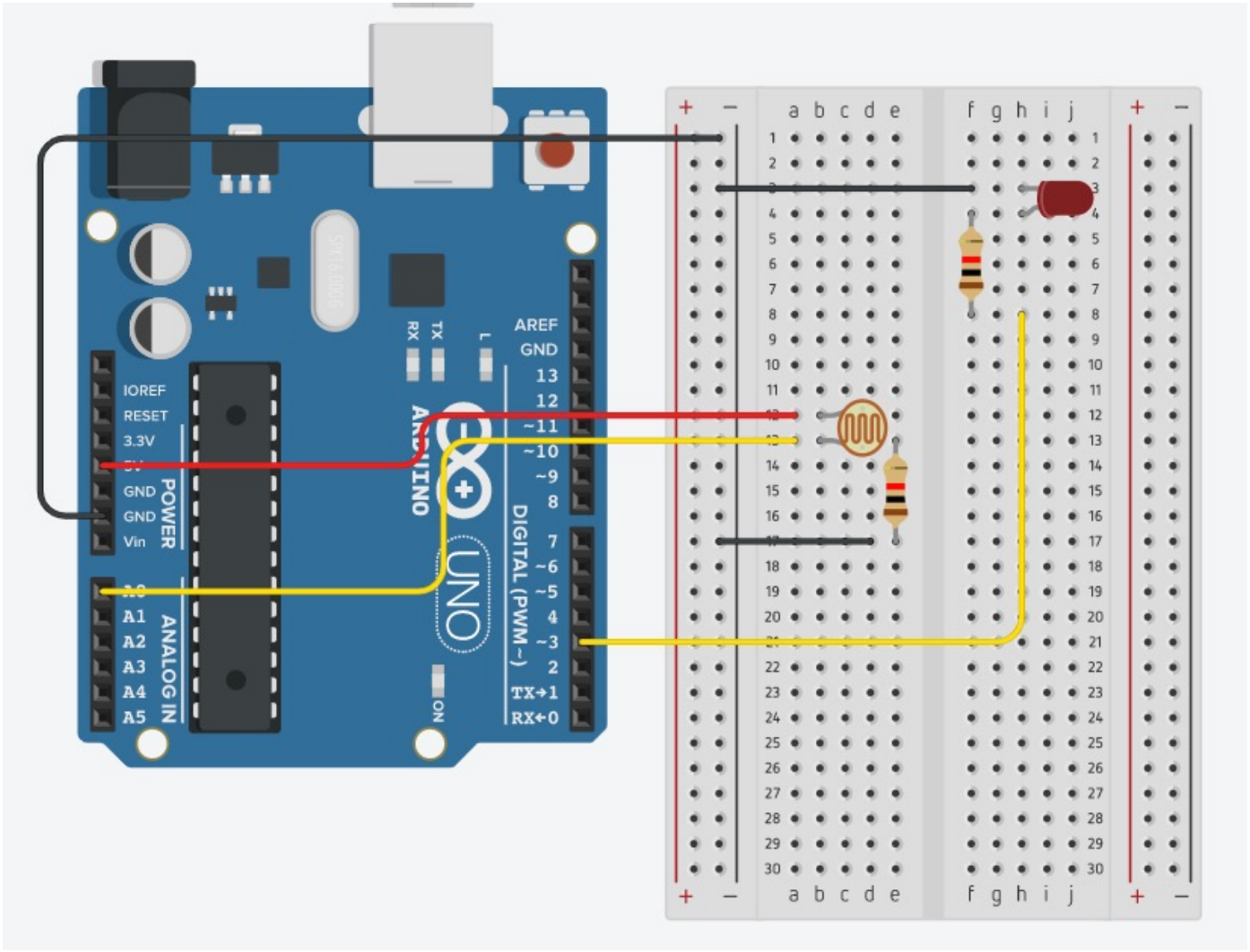
```
// Example: Smart Light with two LEDs

int sensorPin = 0;      // Sensor connected to pin 0

int redLightPin = 3;    // Red LED connected to pin 3
int greenLightPin = 5;  // Green LED connected to pin 5

int threshold = 500;    // An analog value to turn the LED on/off


void setup() {
  Serial.begin(9600);              // Reset the Arduine to communitcate via USB

  pinMode(redLightPin,OUTPUT);     // Set the Red LED (pin 3) to Output
  pinMode(greenLightPin,OUTPUT);   // Set the Green LED (pin 5) to Output
}
```

Two LEDs

```
void loop() {
  int sensorValue = analogRead(sensorPin); // Read the value of the Light Sensor

  Serial.println(sensorValue,DEC);         // Send the light sensor value back to the computer


if (sensorValue < threshold){              // If the sensor value is below the threshold
  digitalWrite(greenLightPin, HIGH);       // set the Green LED to HIGH
  digitalWrite(redLightPin,LOW);           // set the Red LED to LOW
}


if (sensorValue > threshold){              // If the sensor value is above the threshold
  digitalWrite(greenLightPin, LOW);        // set the Green LED to LOW
  digitalWrite(redLightPin, HIGH);         // set the Red LED to HIGH
}
}
```

technocamps

LED Dimmer with Sensor



LED Dimmer with Sensor

```
// Example: Dimmer LED with Sensor

int sensorPin = 0;    // Sensor connected to pin 0
int lightPin = 3;     // LED connected to pin 3

int darkest = 460;    // An analog value of darkness to change the LED value
int lightest = 620;   // An analog value of brightness to change the LED value
```

LED Dimmer with Sensor

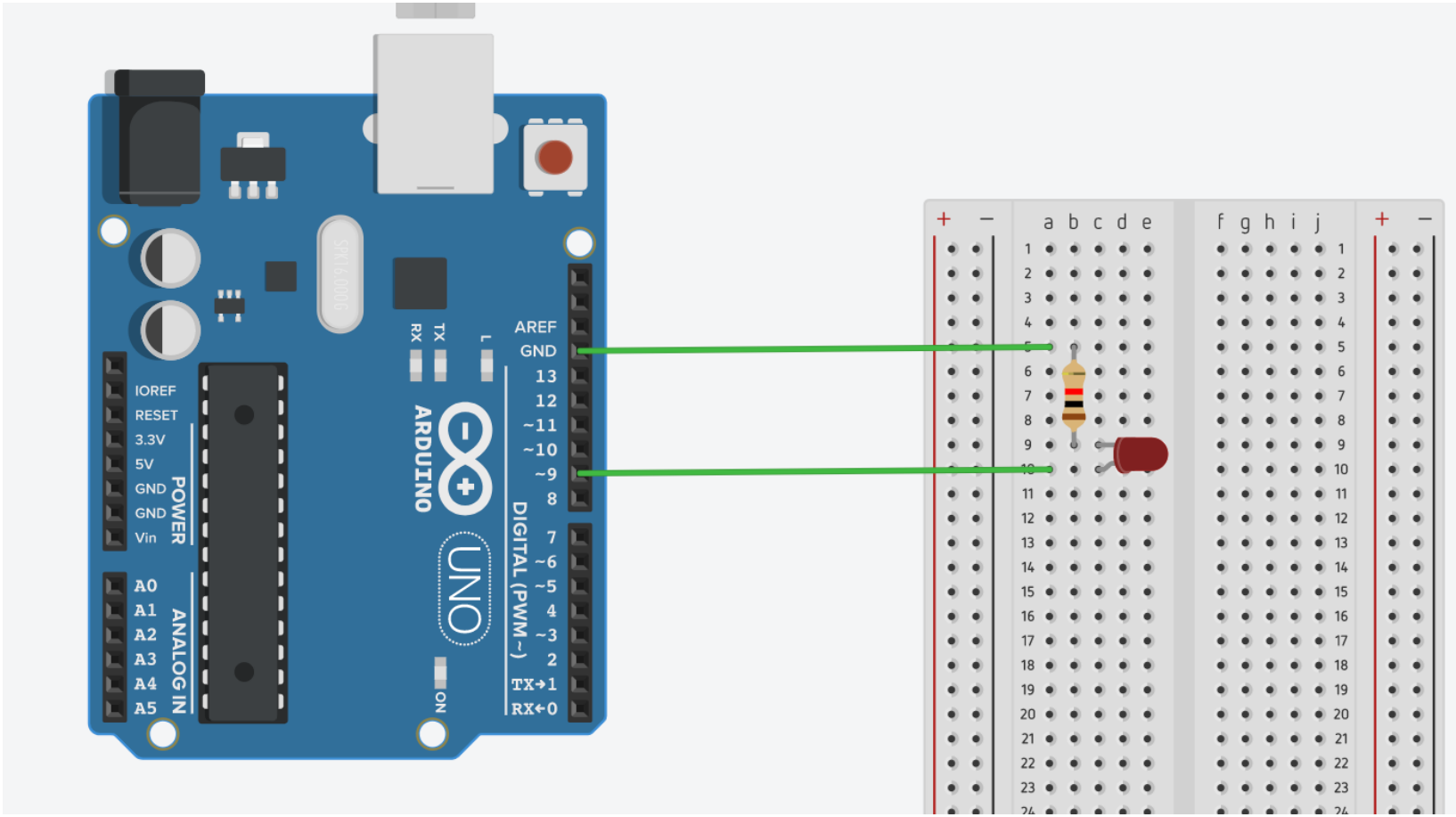
```
void setup() {
  Serial.begin(9600);          // Reset the Arduine to communitcate via USB

  pinMode(lightPin, OUTPUT);  // Set the LED (pin 3) to Output
}


void loop() {
  int sensorValue = analogRead(sensorPin);       // Read the value of the Light Sensor
  Serial.println(sensorValue);                   // Send the light sensor value back to the computer

  int brightness = setBrightness(sensorValue);  // Set the brightness from the sensor value
  analogWrite(lightPin, brightness);             // Write the brightness value to the LED
}
```

LED Dimmer with Sensor

```
int setBrightness(int value) {                          // Create the function setBrightness from the
                                                        // sensor value and convert to digital output

  value = max(value, darkest);                          // Set the worth of value to the
                                                        // highest value within the brakets

  value = min(value, lightest);                         // Set the worth of value to the
                                                        // lowest value within the brakets

  value = map(value, darkest, lightest, 0, 255);        // Create a map to equate the sensor value
                                                        // to the range of values the LED can have

  value = 255 - value;                                  // Change the LED value to the
                                                        // reverse of the sensor value

  return value;                                         // Send the value back out of this function
}
```

Let's Get More Advanced

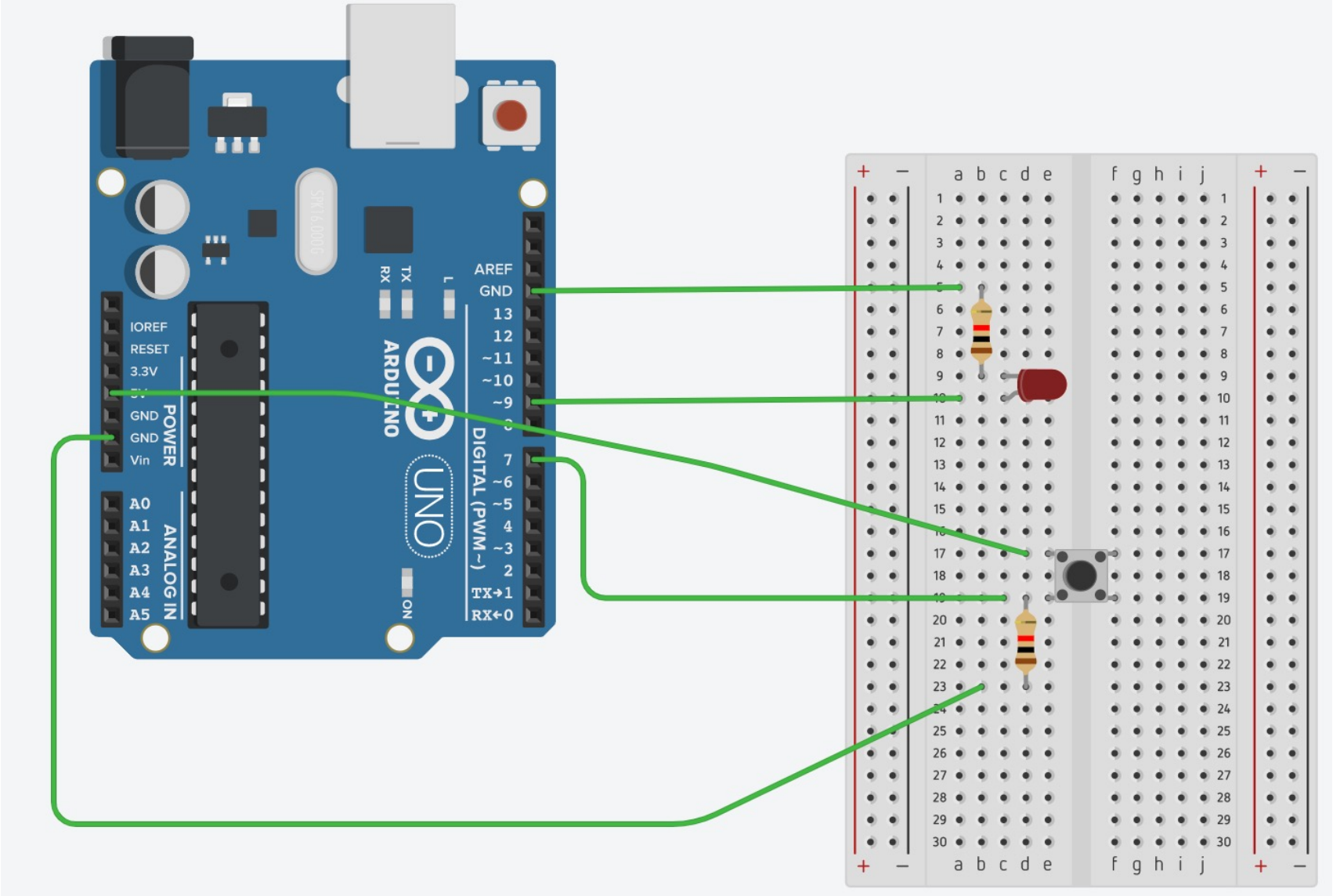
Fading LED


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

The Code

```
// Example 04: Fade an LED in and out like on a sleeping Apple computer

const int LED = 9;         // the pin for the LED
int i = 0;                 // We'll use this to count up and down

void setup()
{
  pinMode(LED, OUTPUT);    // tell Arduino LED is an output
}

void loop()
{

  for (i=0; i<255; i++)    // loop from 0 to 254 (fade in)
  {
    analogWrite(LED,i);    // set the LED brightness
    delay(10);             // Wait 10ms because analogWrite is instantaneous and we would not see any change
  }

  for (i=255; i>0; i--)    // loop from 255 to 1 (fade out)
  {
    analogWrite(LED,i);     // set the LED brightness
    delay(10);              // Wait 10ms
  }

}
```

How Can We Use This For Our Old Lamp Demo?


AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
IOREF
RESET
3.3V
5V
GND
GND
Vin
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
ON
TX
RX
L
UNO
ARDUINO
a b c d e
f g h i j
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30

The Code

```
// Example 05: Turn on LED when the button is pressed and keep it on after it is released including simple de-bouncing.

const int LED = 9;              // the pin for the LED
const int BUTTON = 7;           // inpput pin of the pushbutton

int val = 0;                    // stores the state of the input pin
int old_val = 0;                // stores the previous value of "val"
int state = 0;                  // 0 = LED off while 1 = LED on

int brightness = 128;           // Stores the brightness value
unsigned long startTime = 0;  // when did we begin pressing?

void setup()
{
  pinMode(LED, OUTPUT);       // tell Arduino LED is an output
  pinMode(BUTTON, INPUT);     // tell Arduino BUTTON is an input
}
```

The Code

```
void loop()
{
  val = digitalRead(BUTTON);  // read input value and store it yum, fresh

  // check if there was a transition
  if ((val == HIGH) && (old_val == LOW))
  {
    state = 1 - state;        // change the state from off to on or vice-versa

    startTime = millis();     // millis() is the Arduino clock. It returns how many milliseconds have passed since the board has been reset.

    // (this line remembers when the button was last pressed)
    delay(10);
  }
```

The Code

```
  // check wether the button is being held down
  if ((val == HIGH) && (old_val == HIGH))
  {
    // if the button is held for more than 500ms
    if (state == 1 && (millis() - startTime) > 500)
    {
      brightness++;           // increment brightness by 1
      delay(25);              // delay to avoid brightness going up too fast

      if (brightness > 255)
      {
        // 255 is the max brightness
        brightness = 0;       // if we go over 255 let's go back to 0
      }
    }
  }

  old_val = val;              // val is now old, let's store it

  if ( state == 1)
  {
    analogWrite(LED, brightness);   // turn LED ON at the current brightness level
  }
  else
  {
    analogWrite(LED,0);             // turn the LED off
  }
}
```

Traffic Light System

Use the functions you've learnt while coding today to write a program that runs a traffic light system.

- You can use 3 LEDs (red, amber, green) – set up each LED separately using a resistor for each one.
- Use a sensible sequence within the loop (i.e. red, red and amber, green, amber).
- Attempt to code this sequence first before considering adding a button or sensor!