SOUND INTERACTION WORKSHOP

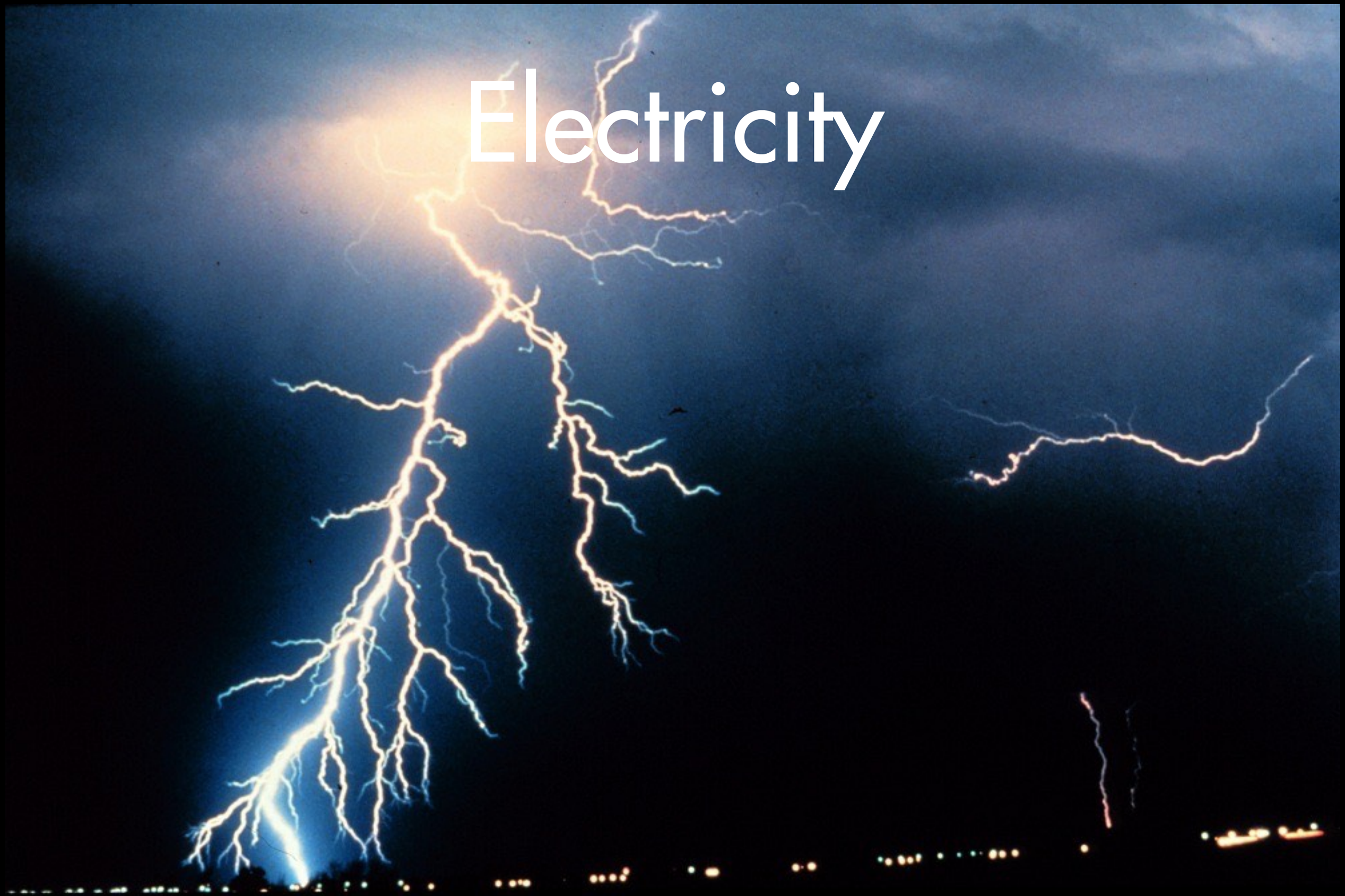
Sensing the world

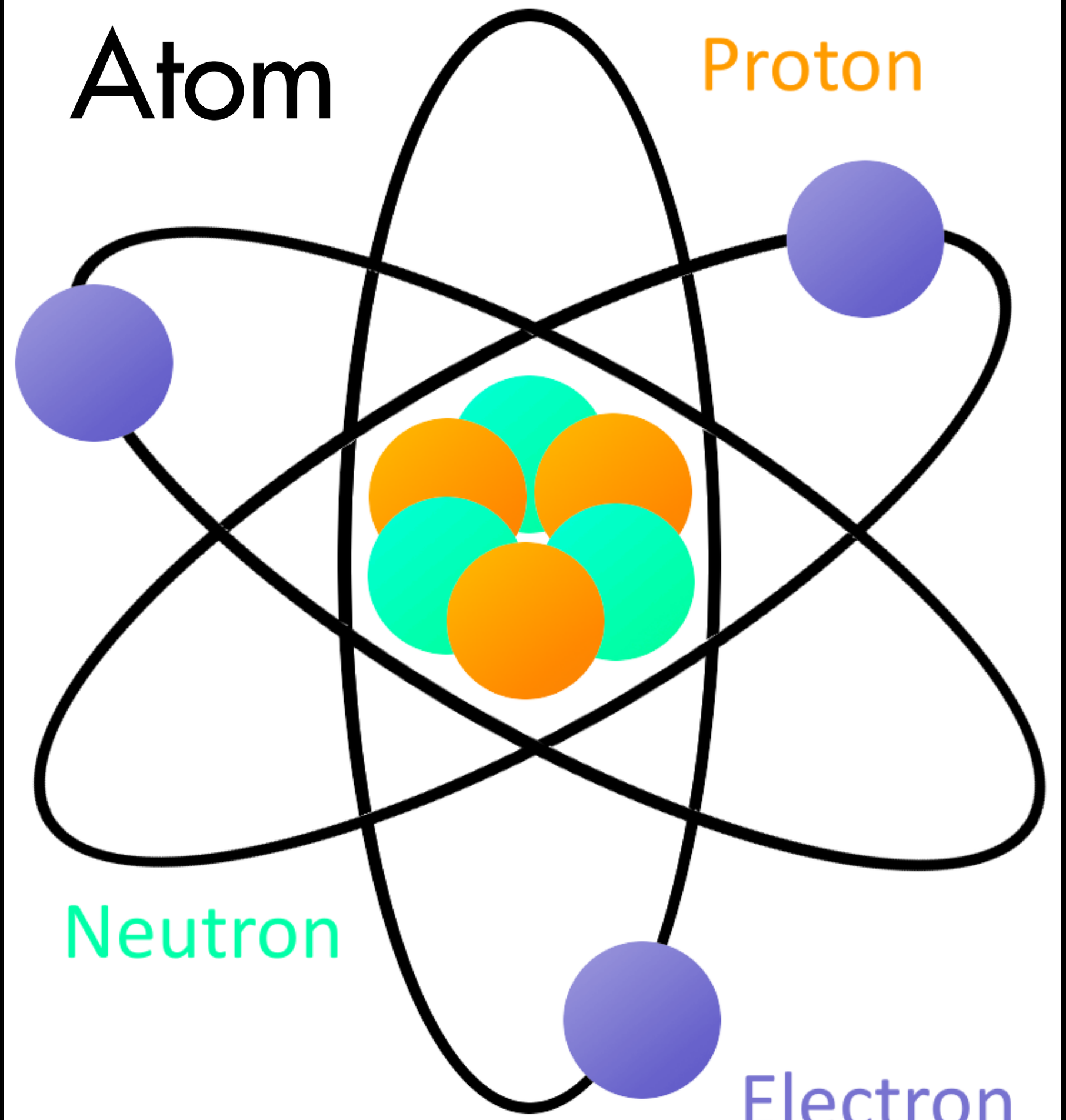
Imanol Gómez
imanolgomez.net
5th May 2017

INTRODUCTION

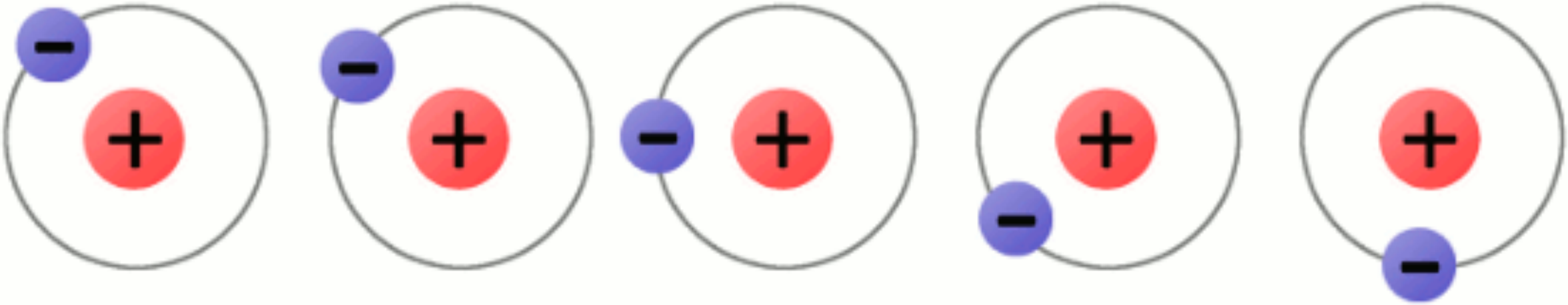
ELECTRONICS

Electricity

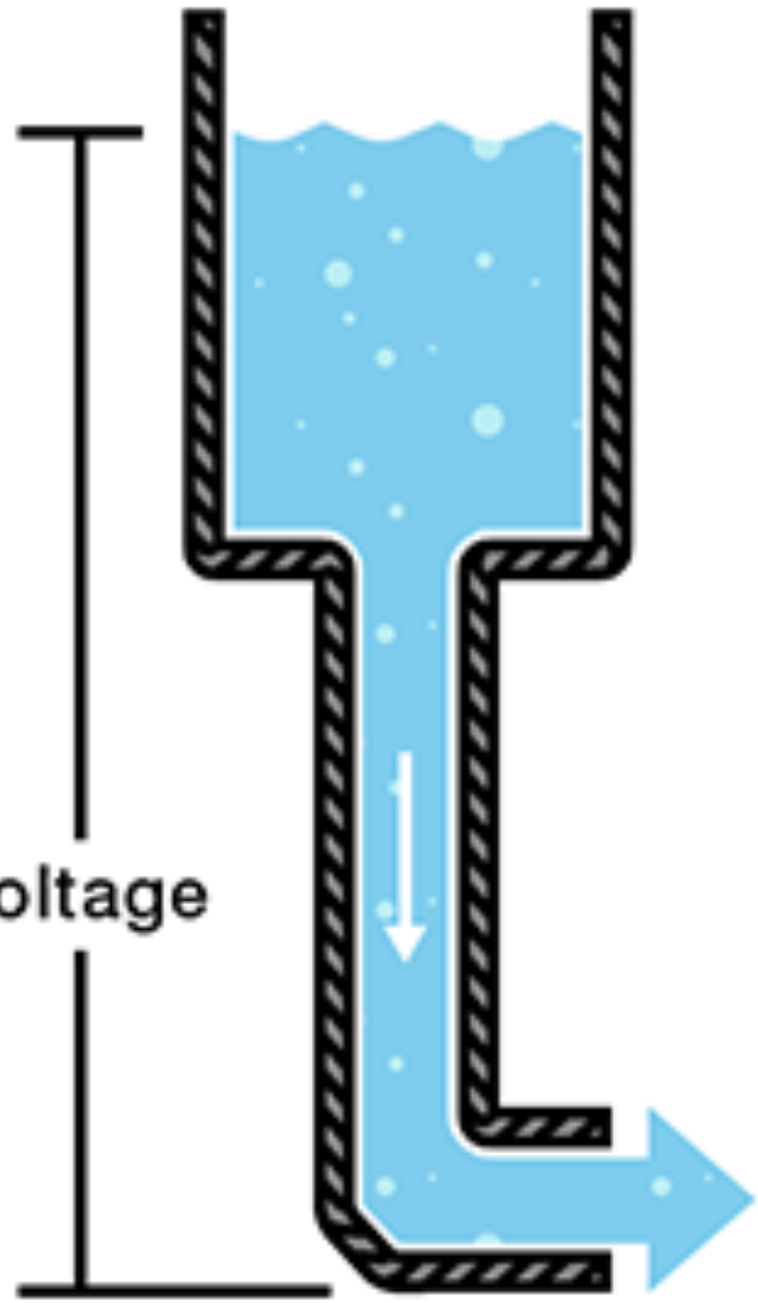



Atom
Proton
Neutron
Electron

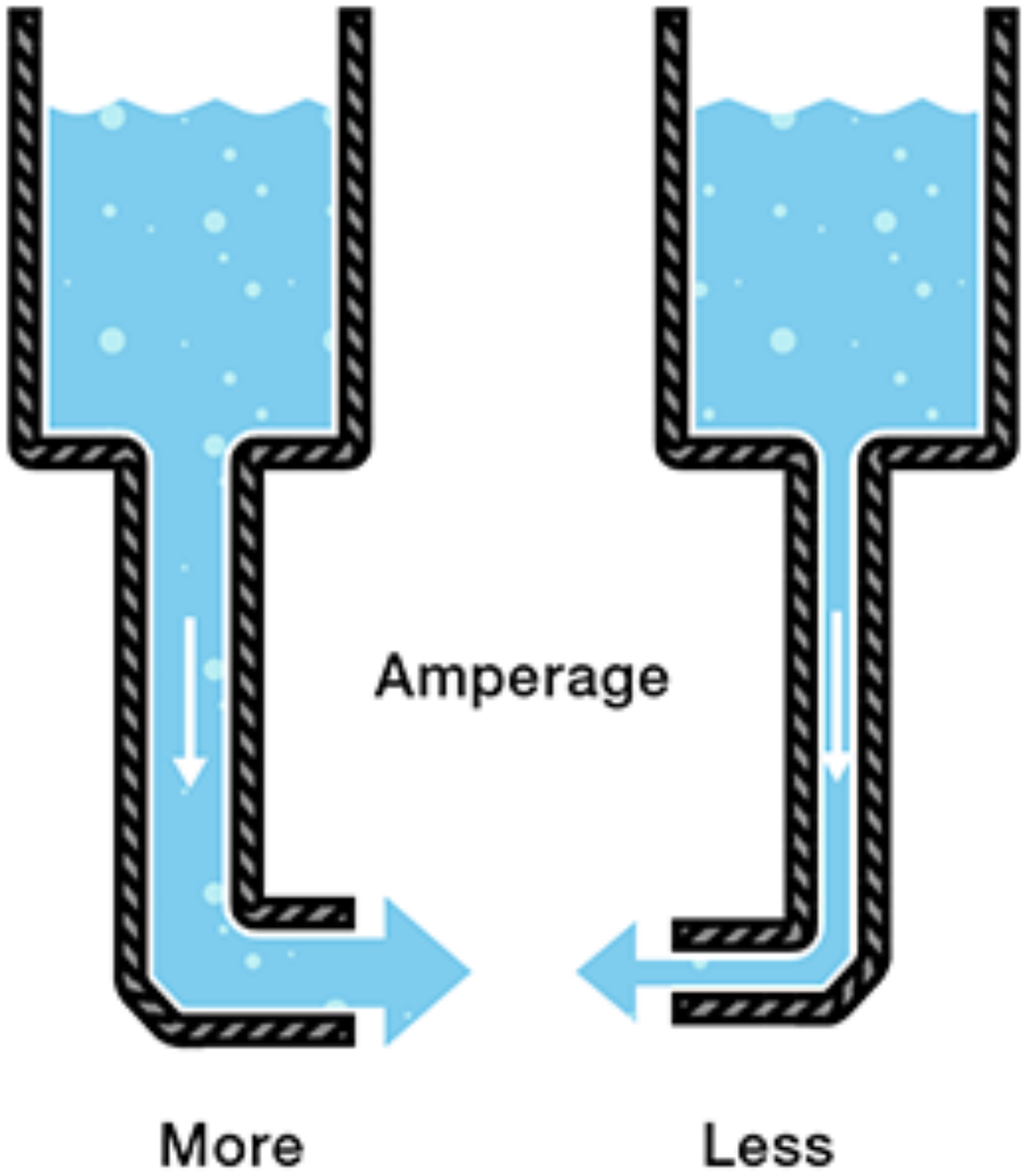
Flowing Charges



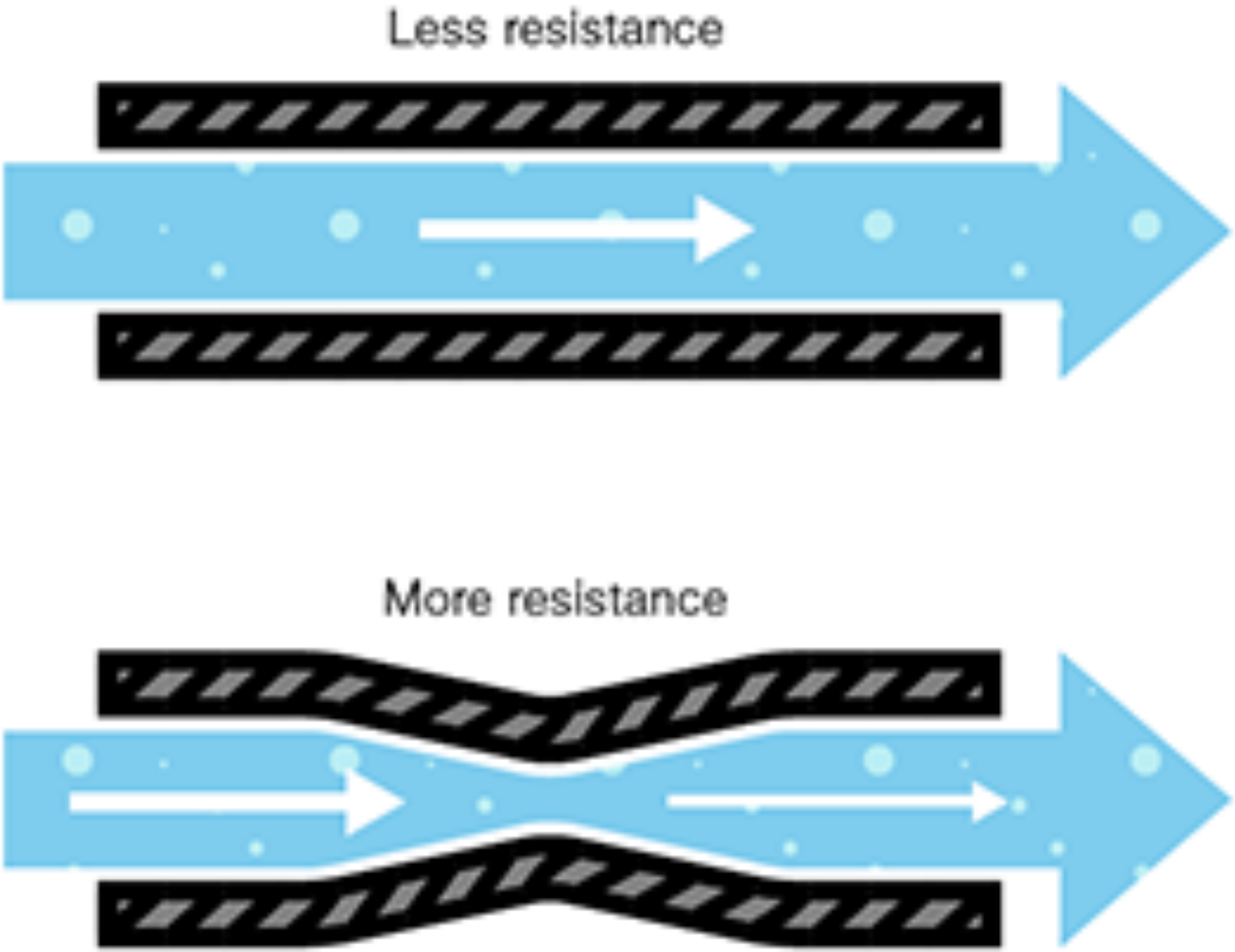
Voltage



Current



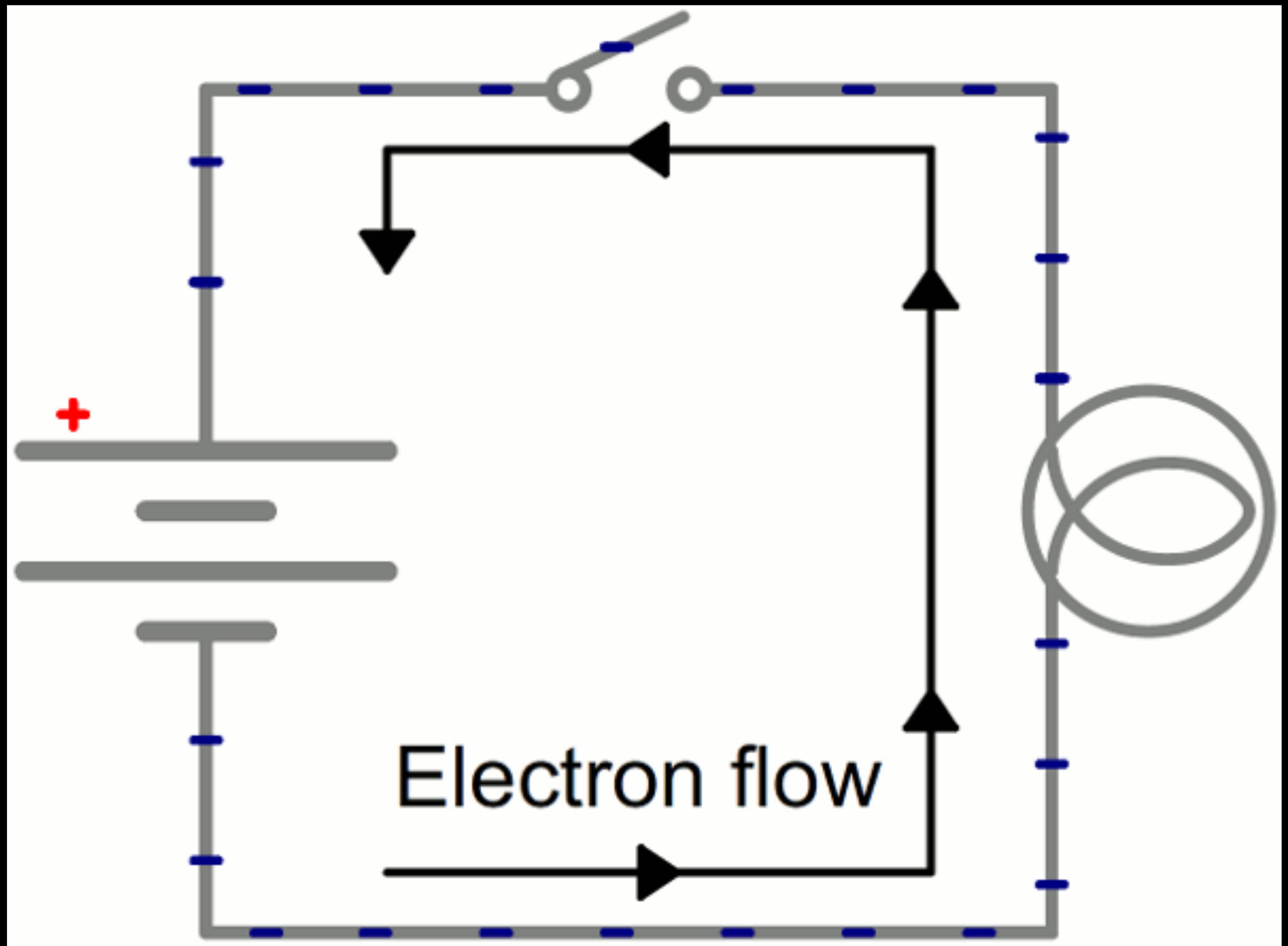
Resistance



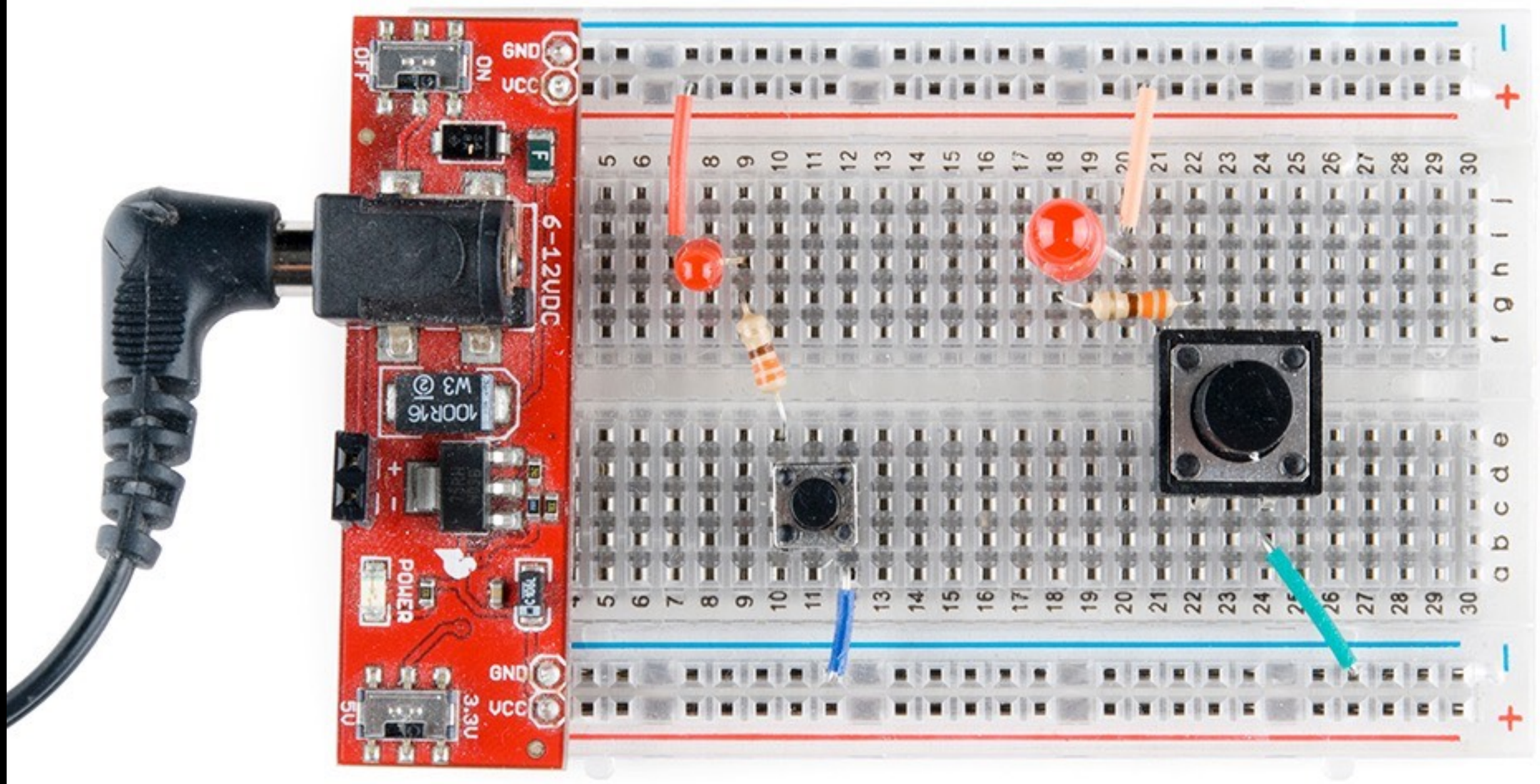
Ohms Law

$$V = I * R$$

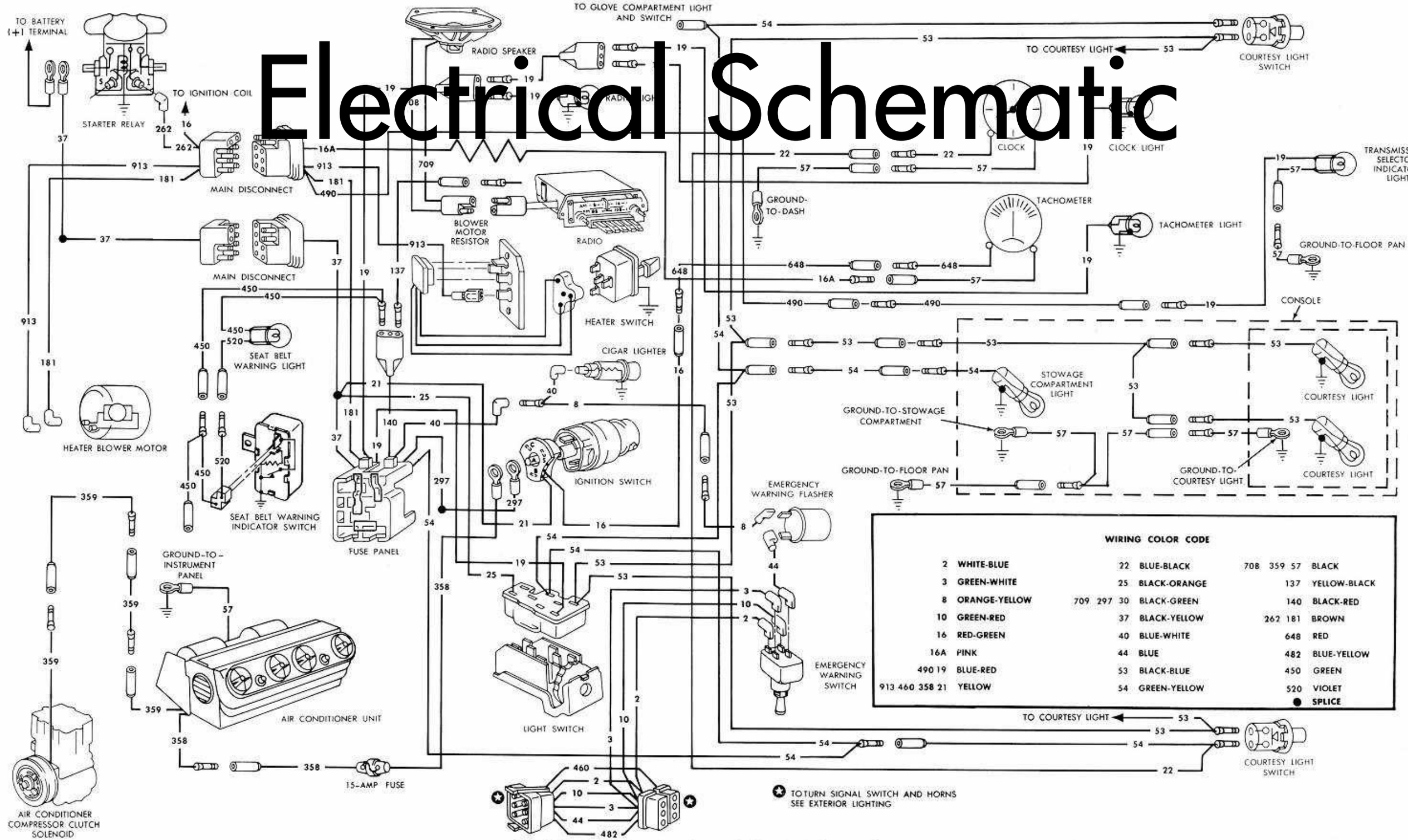
Circuits



Circuits


OFF
ON
GND
VCC
6-12VDC
100R16
W3
POWER
GND
VCC
3.3V
5V
a b c d e f g h i j
+
-

Electrical Schematic

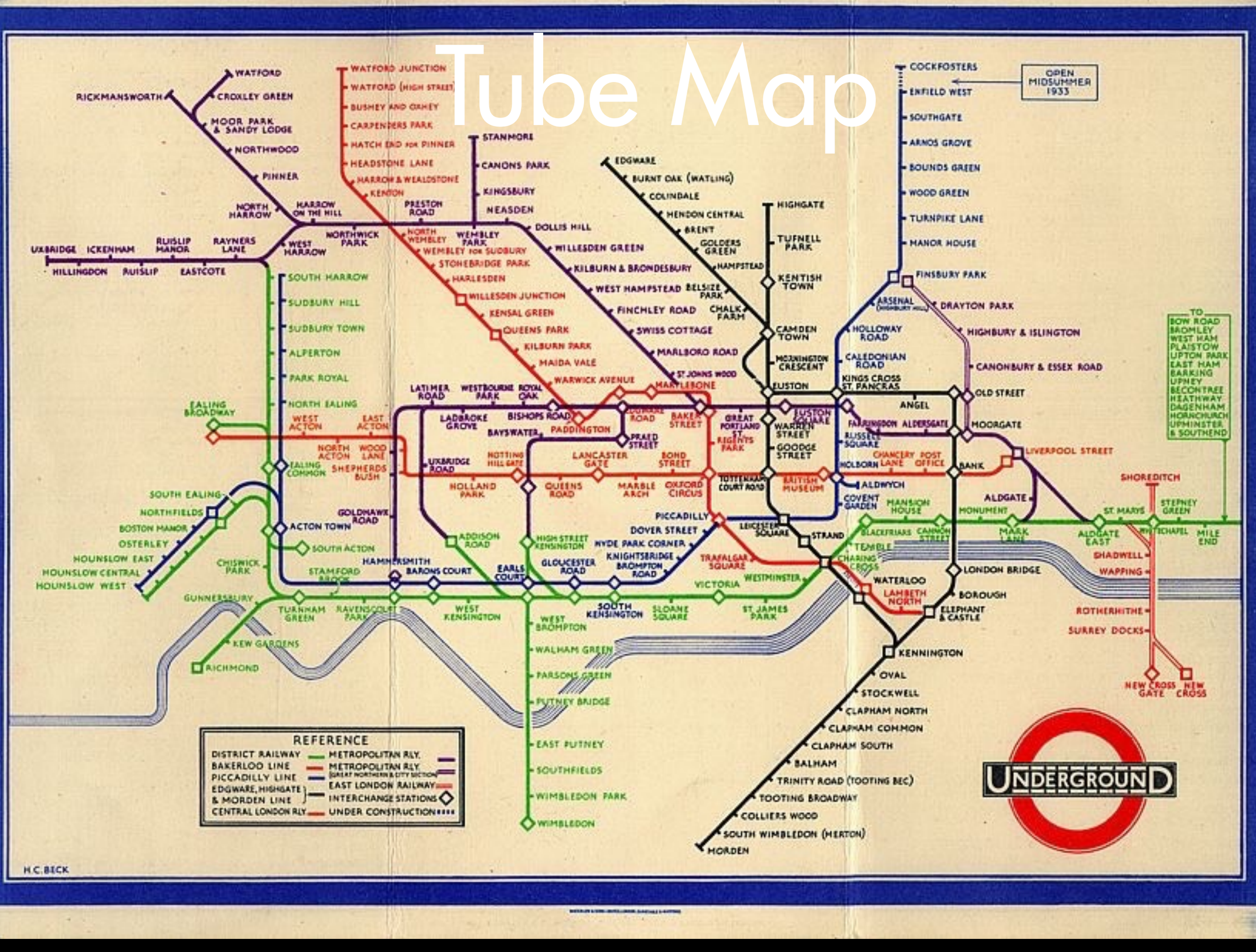


WIRING COLOR CODE

2	WHITE-BLUE	22	BLUE-BLACK	708 359 57	BLACK
3	GREEN-WHITE	25	BLACK-ORANGE	137	YELLOW-BLACK
8	ORANGE-YELLOW	709 297 30	BLACK-GREEN	140	BLACK-RED
10	GREEN-RED	37	BLACK-YELLOW	262 181	BROWN
16	RED-GREEN	40	BLUE-WHITE	648	RED
16A	PINK	44	BLUE	482	BLUE-YELLOW
490 19	BLUE-RED	53	BLACK-BLUE	450	GREEN
913 460 358 21	YELLOW	54	GREEN-YELLOW	520	VIOLET
				●	SPLICE

1966 MUSTANG ACCESSORIES

Tube Map


OPEN MIDSUMMER 1933
COCKFOSTERS
ENFIELD WEST
SOUTHGATE
ARNOS GROVE
BOUNDS GREEN
WOOD GREEN
TURNPIKE LANE
MANOR HOUSE
FINSBURY PARK
ARSENAL
DRAYTON PARK
HIGHBURY & ISLINGTON
CANONBURY & ESSEX ROAD
OLD STREET
MOORGATE
LIVERPOOL STREET
SHOREDITCH
ALDGATE
ALDGATE EAST
STEPNEY GREEN
MILE END
WHITECHAPEL
SHADWELL
WAPPING
ROTHERHITHE
SURREY DOCKS
NEW CROSS GATE
NEW CROSS
TO BOW ROAD BROMLEY WEST HAM PLAISTOW UPTON PARK EAST HAM BARKING UPNEY BECONTREE HEATHWAY DAGENHAM HORNCHURCH UPMINSTER & SOUTHEND
HIGHGATE
TUFNELL PARK
KENTISH TOWN
CAMDEN TOWN
MORNINGTON CRESCENT
EUSTON
KINGS CROSS ST. PANCRAS
ANGEL
FARRINGDON
ALDERSGATE
RUSSELL SQUARE
HOLBORN
CHANCERY LANE
POST OFFICE
BANK
MONUMENT
LONDON BRIDGE
BOROUGH
ELEPHANT & CASTLE
KENNINGTON
OVAL
STOCKWELL
CLAPHAM NORTH
CLAPHAM COMMON
CLAPHAM SOUTH
BALHAM
TRINITY ROAD (TOOTING BEC)
TOOTING BROADWAY
COLLIERS WOOD
SOUTH WIMBLEDON (MERTON)
MORDEN
HOLLOWAY ROAD
CALEDONIAN ROAD
BLACKFRIARS
CANNON STREET
MANSION HOUSE
COVENT GARDEN
ALDWYCH
BRITISH MUSEUM
TOTTENHAM COURT ROAD
GOODGE STREET
WARREN STREET
GREAT PORTLAND ST.
REGENTS PARK
EUSTON SQUARE
EDGWARE
BURNT OAK (WATLING)
COLINDALE
HENDON CENTRAL
BRENT
GOLDERS GREEN
HAMPSTEAD
BELSIZE PARK
CHALK FARM
STANMORE
CANONS PARK
KINGSBURY
NEASDEN
DOLLIS HILL
WILLESDEN GREEN
KILBURN & BRONDESBURY
WEST HAMPSTEAD
FINCHLEY ROAD
SWISS COTTAGE
ST JOHNS WOOD
MARLBORO ROAD
MARYLEBONE
BAKER STREET
REGENTS PARK
BOND STREET
OXFORD CIRCUS
PICCADILLY
DOVER STREET
HYDE PARK CORNER
KNIGHTSBRIDGE
BROMPTON ROAD
LEICESTER SQUARE
STRAND
TEMPLE
CHARING CROSS
TRAFALGAR SQUARE
WATERLOO
LAMBETH NORTH
VICTORIA
WESTMINSTER
SLOANE SQUARE
SOUTH KENSINGTON
ST JAMES PARK
WEST KENSINGTON
EARLS COURT
GLOUCESTER ROAD
HIGH STREET KENSINGTON
ADDISON ROAD
HAMMERSMITH
BARONS COURT
STAMFORD BROOK
CHISWICK PARK
GUNNERSBURY
TURNHAM GREEN
RAVENSCOURT PARK
KEW GARDENS
RICHMOND
WEST BROMPTON
WALHAM GREEN
PARSONS GREEN
PUTNEY BRIDGE
EAST PUTNEY
SOUTHFIELDS
WIMBLEDON PARK
WIMBLEDON
SOUTH EALING
NORTHFIELDS
BOSTON MANOR
OSTERLEY
HOUNSLOW EAST
HOUNSLOW CENTRAL
HOUNSLOW WEST
ACTON TOWN
SOUTH ACTON
GOLDHAWK ROAD
EALING BROADWAY
EALING COMMON
NORTH ACTON
WEST ACTON
EAST ACTON
WOOD LANE
SHEPHERDS BUSH
UXBRIDGE ROAD
HOLLAND PARK
NOTTING HILL GATE
LANCASTER GATE
QUEENS ROAD
MARBLE ARCH
LADBROKE GROVE
WESTBOURNE PARK
ROYAL OAK
BAYSWATER
PADDINGTON
PRAED STREET
LATIMER ROAD
WARWICK AVENUE
MAIDA VALE
KILBURN PARK
QUEENS PARK
KENSAL GREEN
WILLESDEN JUNCTION
HARLESDEN
STONEBRIDGE PARK
WEMBLEY for SUDBURY
NORTH WEMBLEY
WEMBLEY PARK
PRESTON ROAD
HARROW & WEALDSTONE
KENTON
HEADSTONE LANE
HATCH END for PINNER
CARPENDERS PARK
BUSHEY AND OXHEY
WATFORD (HIGH STREET)
WATFORD JUNCTION
NORTHWICK PARK
WEST HARROW
NORTH HARROW
HARROW ON THE HILL
PINNER
NORTHWOOD
MOOR PARK & SANDY LODGE
CROXLEY GREEN
WATFORD
RICKMANSWORTH
UXBRIDGE
ICKENHAM
RUISLIP MANOR
RUISLIP
RAYNERS LANE
EASTCOTE
HILLINGDON
SOUTH HARROW
SUDBURY HILL
SUDBURY TOWN
ALPERTON
PARK ROYAL
NORTH EALING
REFERENCE
DISTRICT RAILWAY
BAKERLOO LINE
PICCADILLY LINE
EDGWARE, HIGHGATE & MORDEN LINE
CENTRAL LONDON RLY
METROPOLITAN RLY.
METROPOLITAN RLY (GREAT NORTHERN & CITY SECTION)
EAST LONDON RAILWAY
INTERCHANGE STATIONS
UNDER CONSTRUCTION
UNDERGROUND
H.C. BECK

Repository

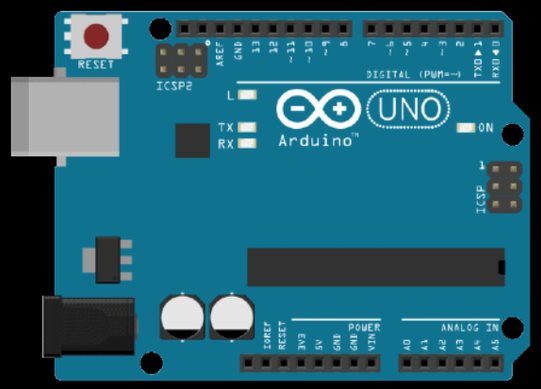
https://github.com/ImanolGo/SoundInteractionWorkshop

ARDUINO

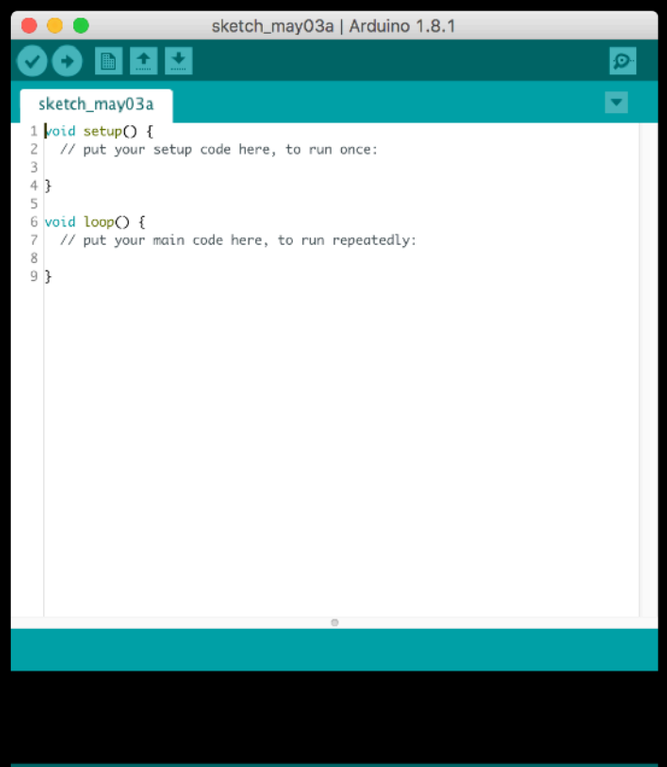

TM
ARDUINO

History

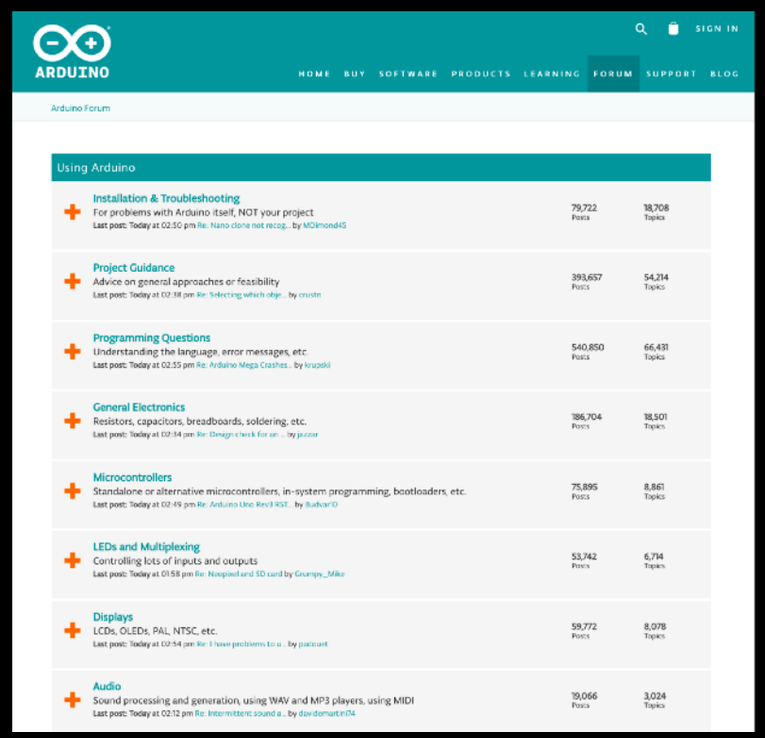
Hardware



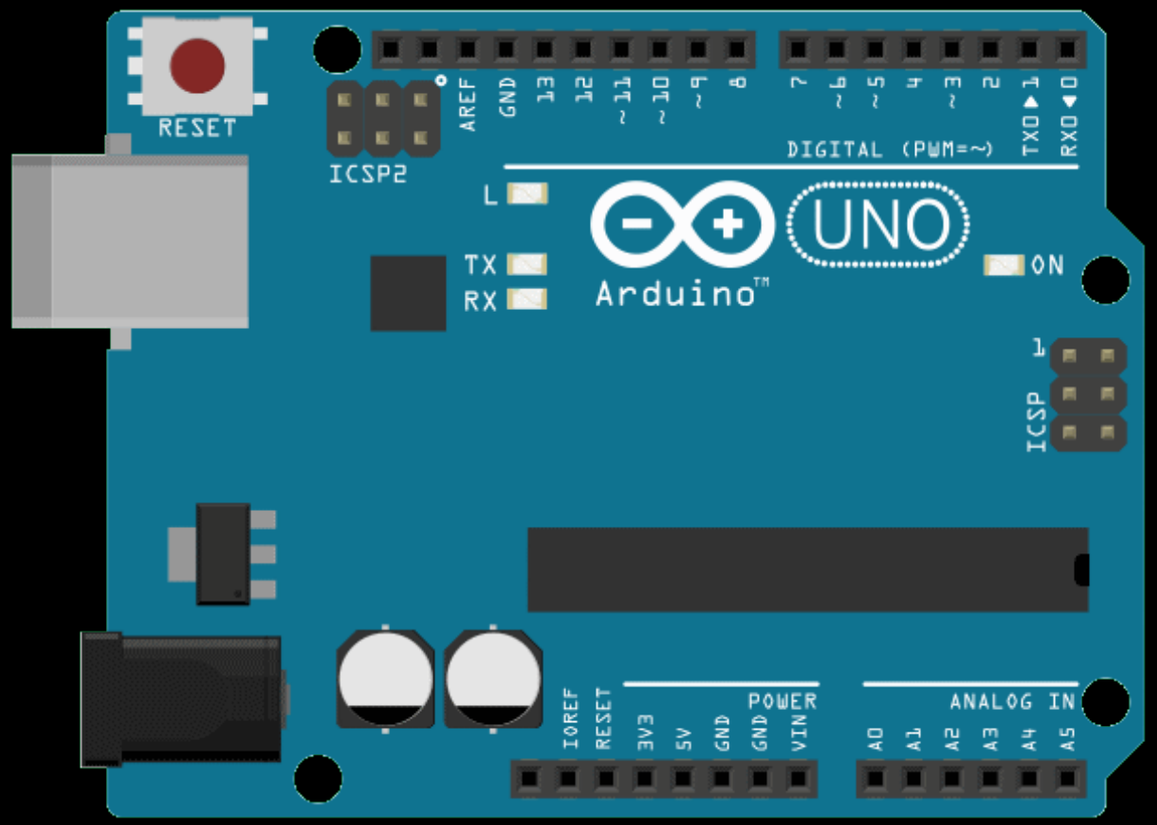
Software



Community



Arduino UNO


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM=~)
L
TX
RX
UNO
Arduino™
ON
ICSP
POWER
ANALOG IN
IOREF
RESET
3V3
5V
GND
GND
VIN
A0
A1
A2
A3
A4
A5

Boards

Arduino Uno

Arduino Leonardo

Arduino Due

Arduino Yún

Arduino Tre

Arduino Micro

Arduino Robot

Arduino Esplora

Arduino Mega ADK

Arduino Ethernet

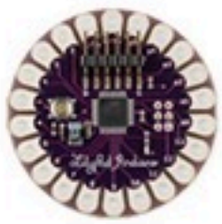
Arduino Mega 2560

Arduino Mini

LilyPad Arduino USB

LilyPad Arduino Simple

LilyPad Arduino SimpleSnap

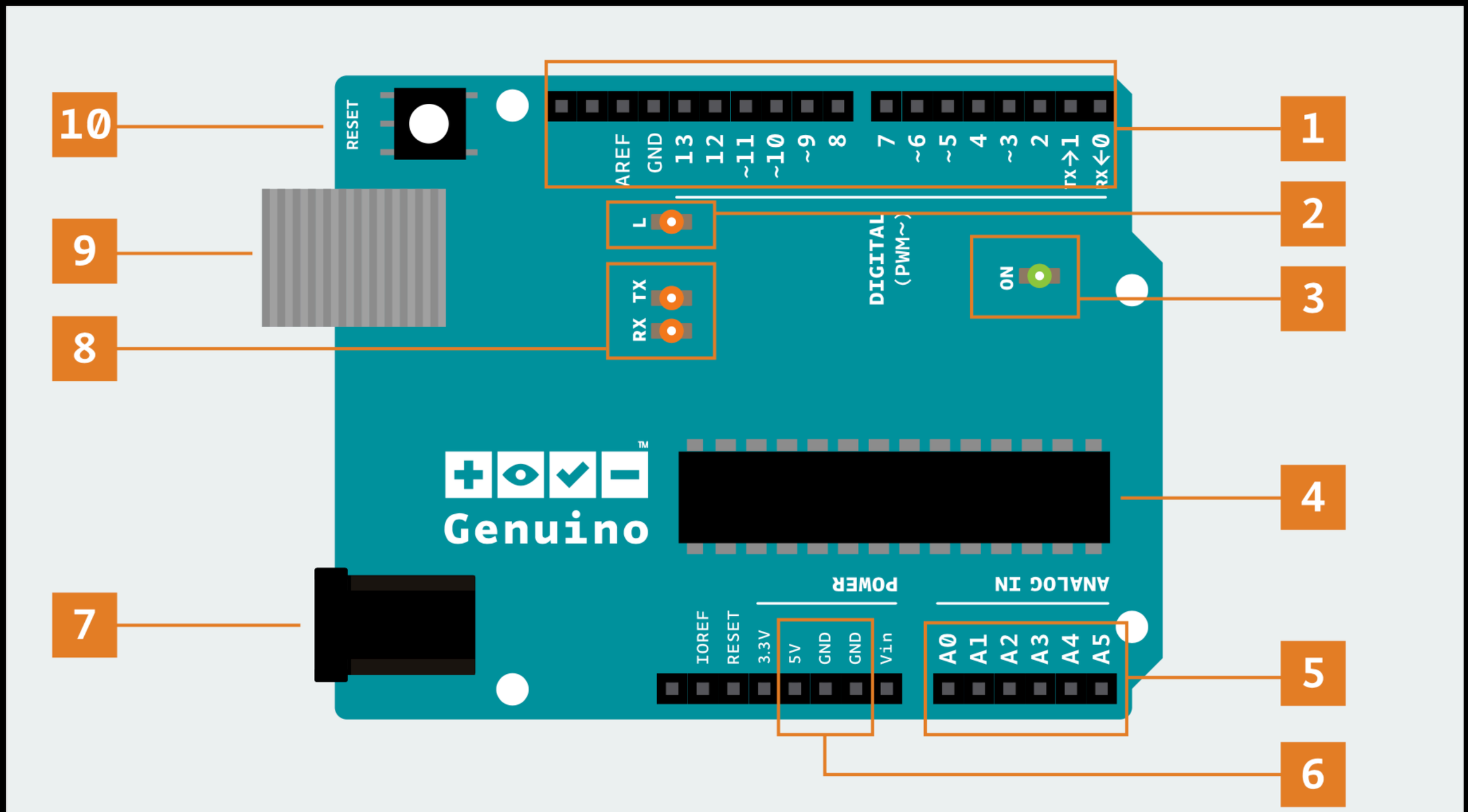


LilyPad Arduino

Arduino Nano

Arduino Pro Mini

What's on the board?



Download Software

ARDUINO 1.8.2

The open-source Arduino Software (IDE) makes it easy to write code and upload it to the board. It runs on Windows, Mac OS X, and Linux. The environment is written in Java and based on Processing and other open-source software.
This software can be used with any Arduino board.
Refer to the Getting Started page for Installation instructions.

Windows Installer
Windows ZIP file for non admin install

Windows app Get

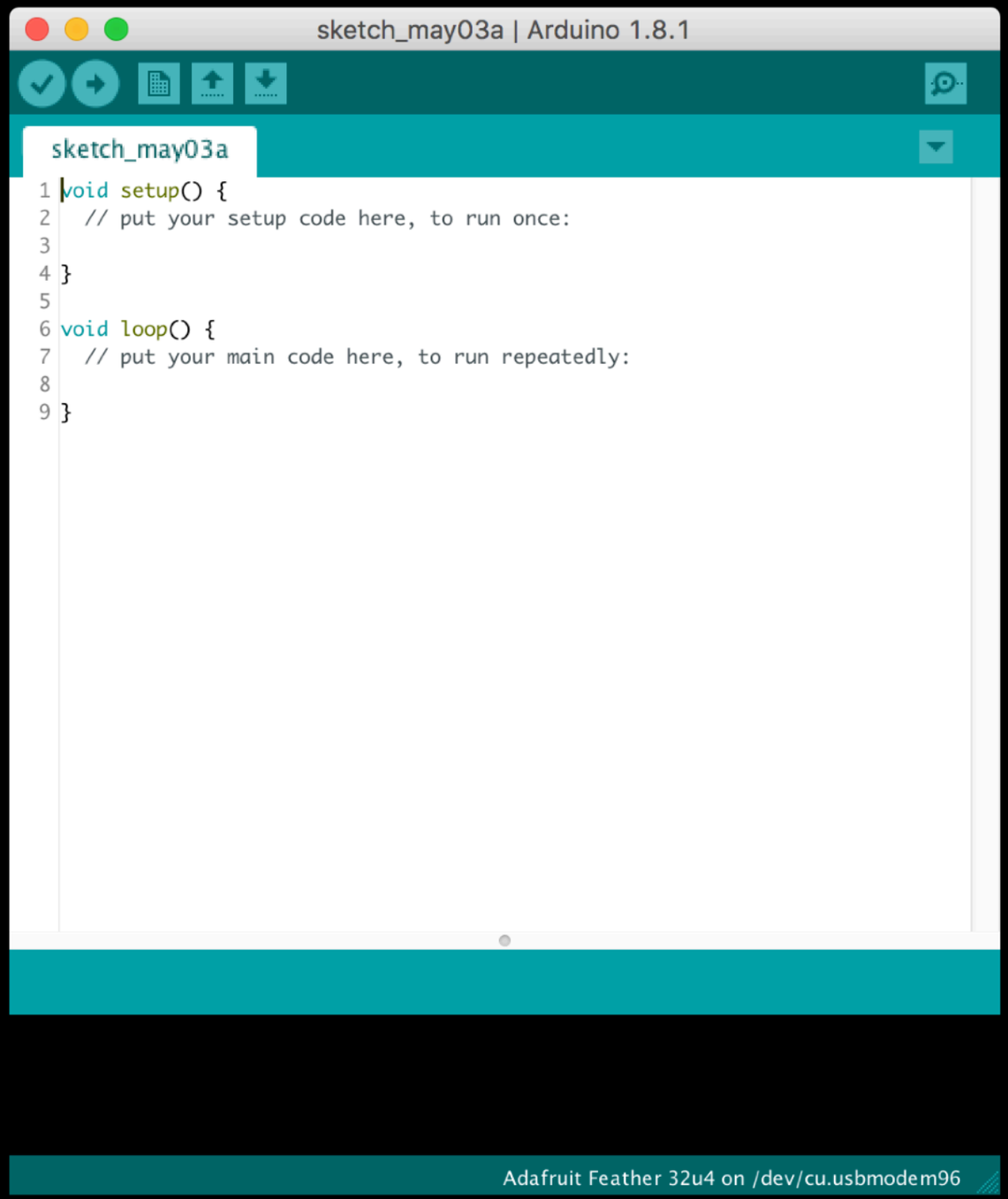
Mac OS X 10.7 Lion or newer

Linux 32 bits
Linux 64 bits
Linux ARM

Release Notes
Source Code
Checksums (sha512)

https://www.arduino.cc/en/Main/Software

Arduino Software



Arduino Software

Verify

Checks your code for errors compiling it.

Upload

Compiles your code and uploads it to the configured board. See uploading below for details.

Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer"

New

Creates a new sketch.

Open

Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the **File** | **Sketchbook** menu instead.

Save

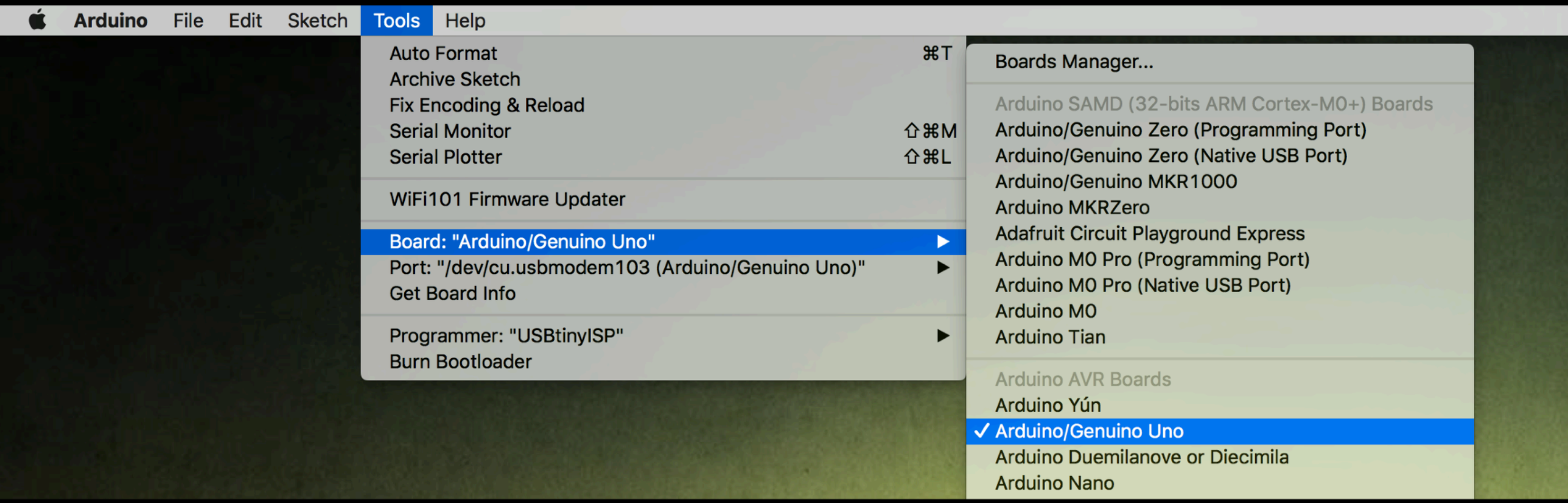
Saves your sketch.

Serial Monitor

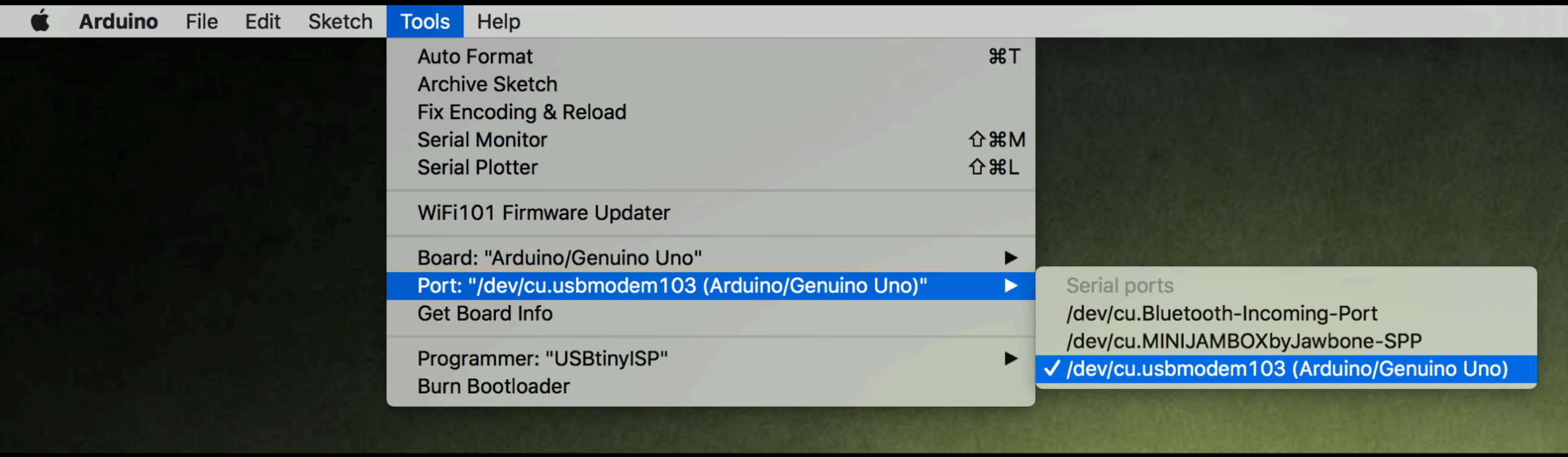
Opens the serial monitor.

Connect the board

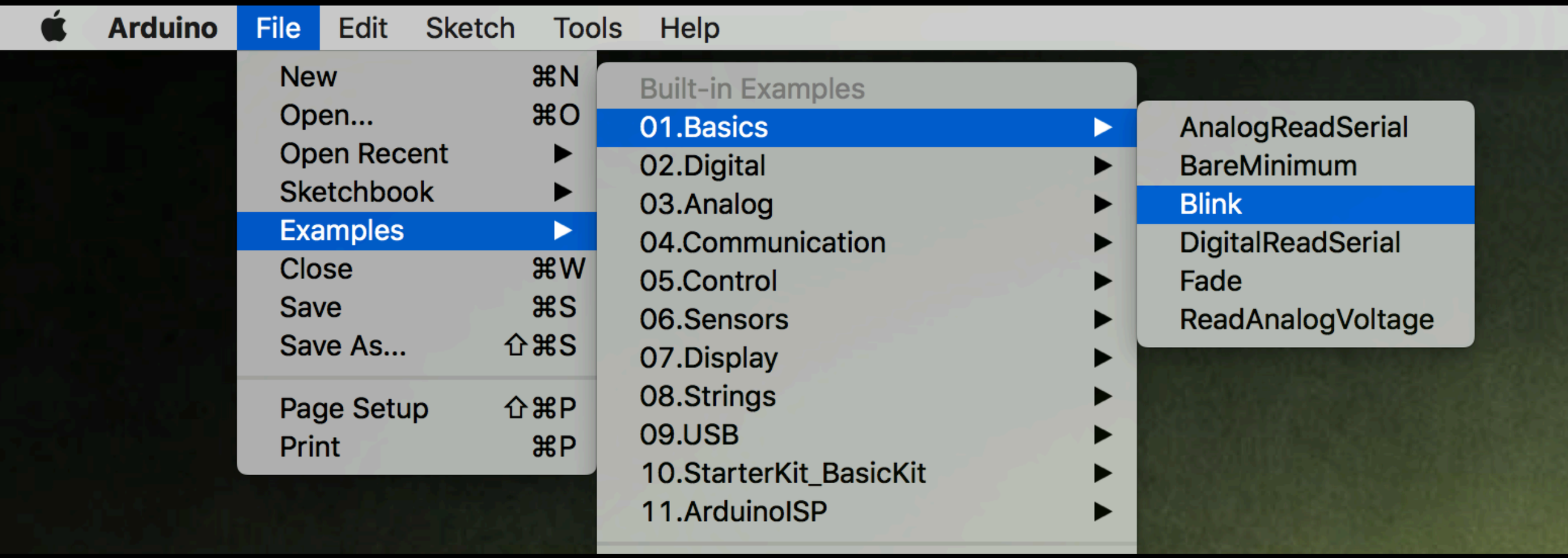
Select Board Type



Select Correct Serial Port



Open Blink Sketch



Upload

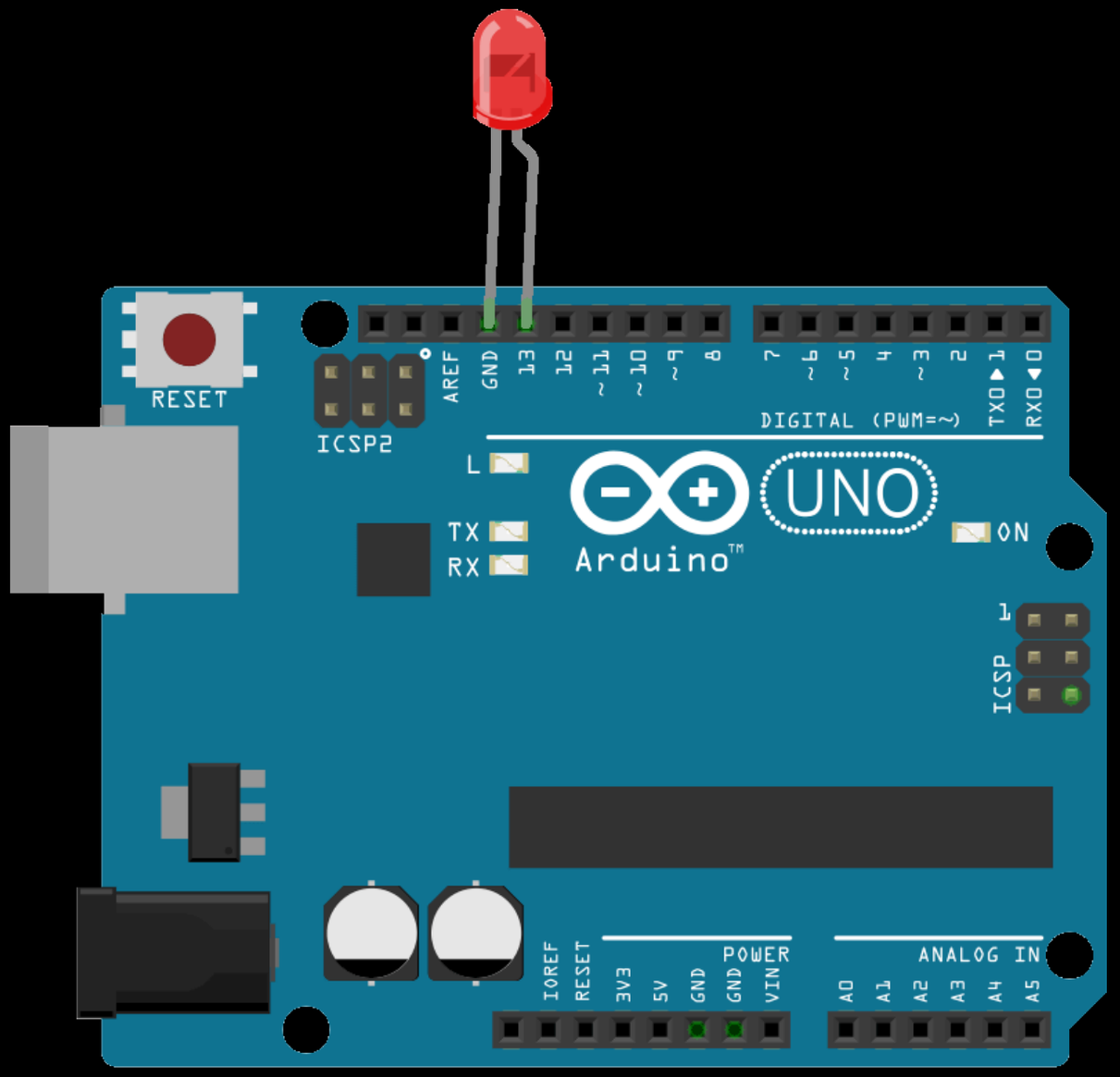

Blink | Arduino 1.8.1
Blink
/*
Blink
Turns on an LED on for one second, then off for one second, repeatedly.
Most Arduinos have an on-board LED you can control. On the UNO, MEGA and ZERO
it is attached to digital pin 13, on MKR1000 on pin 6. LED_BUILTIN is set to
the correct LED pin independent of which board is used.
If you want to know what pin the on-board LED is connected to on your Arduino model, check
the Technical Specs of your board at https://www.arduino.cc/en/Main/Products
This example code is in the public domain.
modified 8 May 2014
by Scott Fitzgerald
modified 2 Sep 2016
by Arturo Guadalupi
modified 8 Sep 2016
by Colby Newman
*/
// the setup function runs once when you press reset or power the board
void setup() {
// initialize digital pin LED_BUILTIN as an output.
pinMode(LED_BUILTIN, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
digitalWrite(LED_BUILTIN, HIGH); // turn the LED on (HIGH is the voltage level)
delay(1000); // wait for a second
digitalWrite(LED_BUILTIN, LOW); // turn the LED off by making the voltage LOW
delay(1000); // wait for a second
}

Watch!

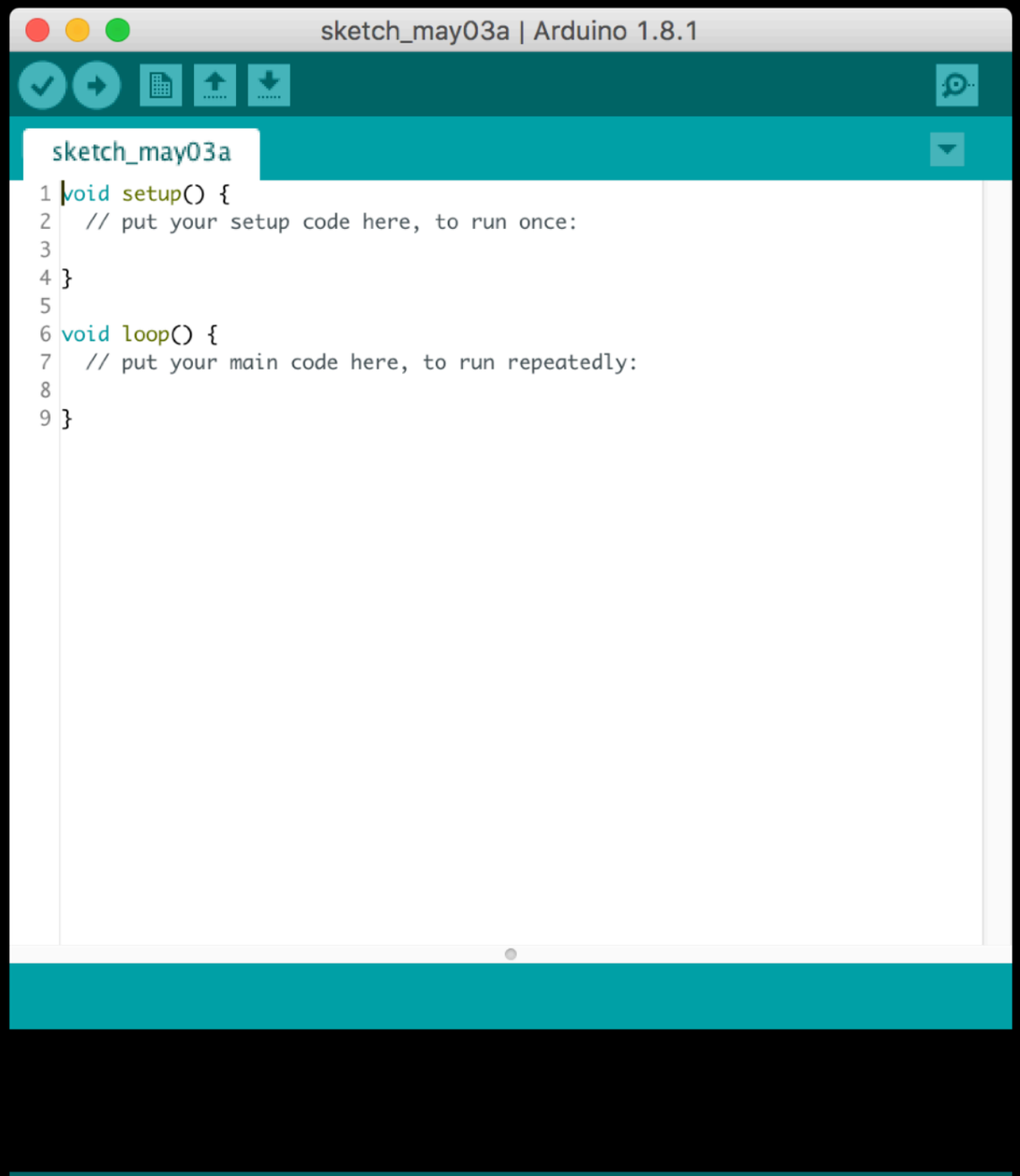

DIGITAL (PWM~)
UNO
ON
ARDUINO
MADE IN ITALY
ICSP
POWER
ANALOG IN
ARDUINO.CC

How long does Blink run for?

Experiment


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM=~)
L
TX
RX
UNO
Arduino™
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN

Program Structure



Statement

```
// initialize digital pin 13 as an output.
pinMode(13, OUTPUT);
```

Function

Function name	(	Inputs & Details Separated by commas	)	;
pinMode	(	13, OUTPUT	)	;

Inside the code

Blink | Arduino 1.8.1

Blink §

```
1
2 // the setup function runs once when you press reset or power the board
3 void setup() {
4   // initialize digital pin LED_BUILTIN as an output.
5   pinMode(LED_BUILTIN, OUTPUT);
6 }
7
8 // the loop function runs over and over again forever
9 void loop() {
10   digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage level)
11   delay(1000);                       // wait for a second
12   digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltage LOW
13   delay(1000);                       // wait for a second
14 }
15
16
17
```

17 Arduino/Genuino Uno on /dev/cu.usbmodem103

Changing the delay

```
delay(500);
```

Exercises!

- Exercise 1: Modify the code so that the light is on for 100 milliseconds and off for 900 milliseconds. This makes for a nice once-per-second second timer.

- Exercise 2: Modify the code so that the light is on for 50 milliseconds and off for 50 milliseconds. What happens?

- Exercise 3: Modify the code so that the light is on for 10 ms (the shorthand for milliseconds) and off for 10 ms. What happens?

Project: Dubstep LED Blinker

Variable

```
int inputVariable1;
int inputVariable2 = 0;     // both are correct
```

Random Blink

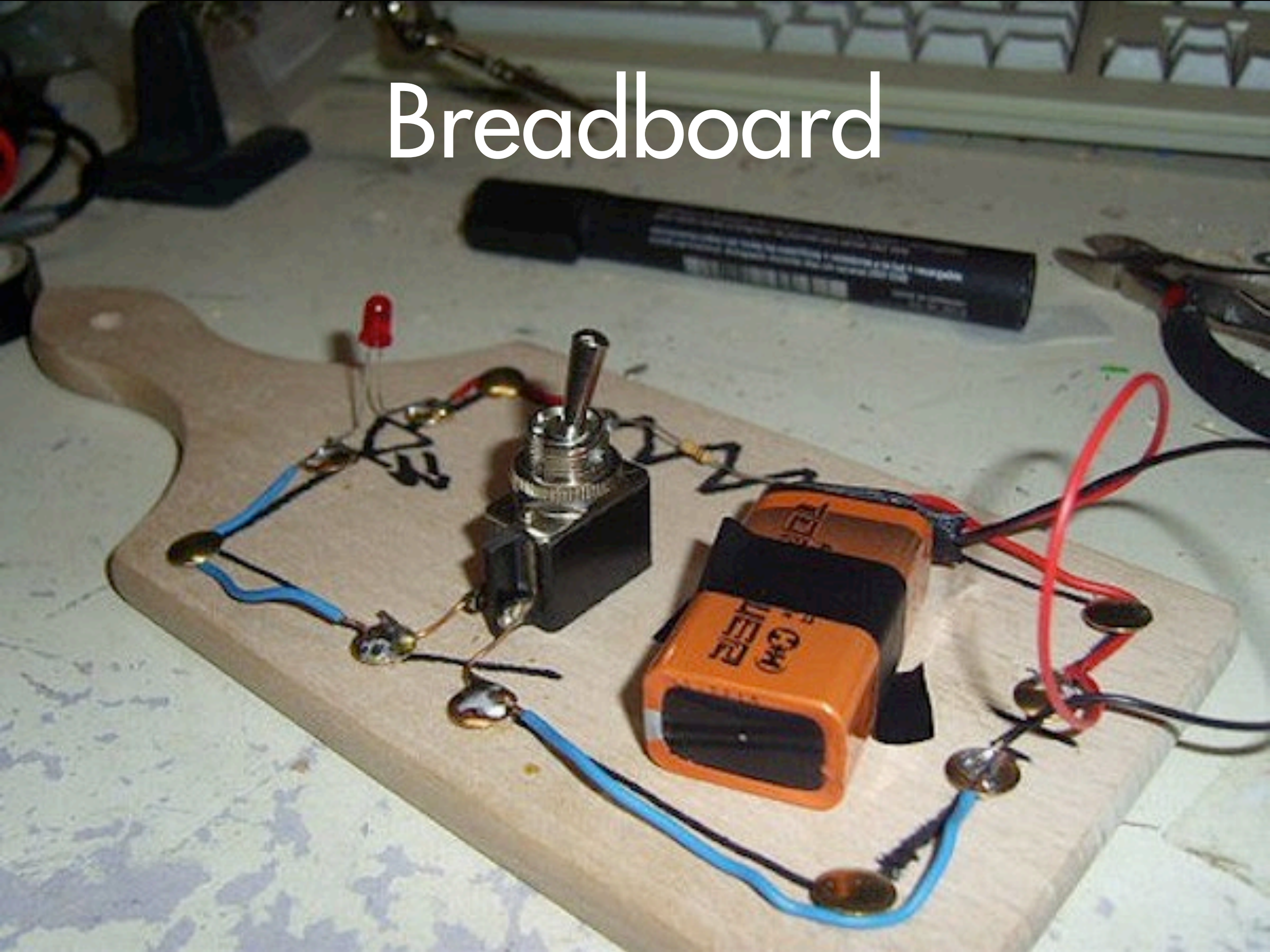
RandomLedBlinker | Arduino 1.8.1

RandomLedBlinker

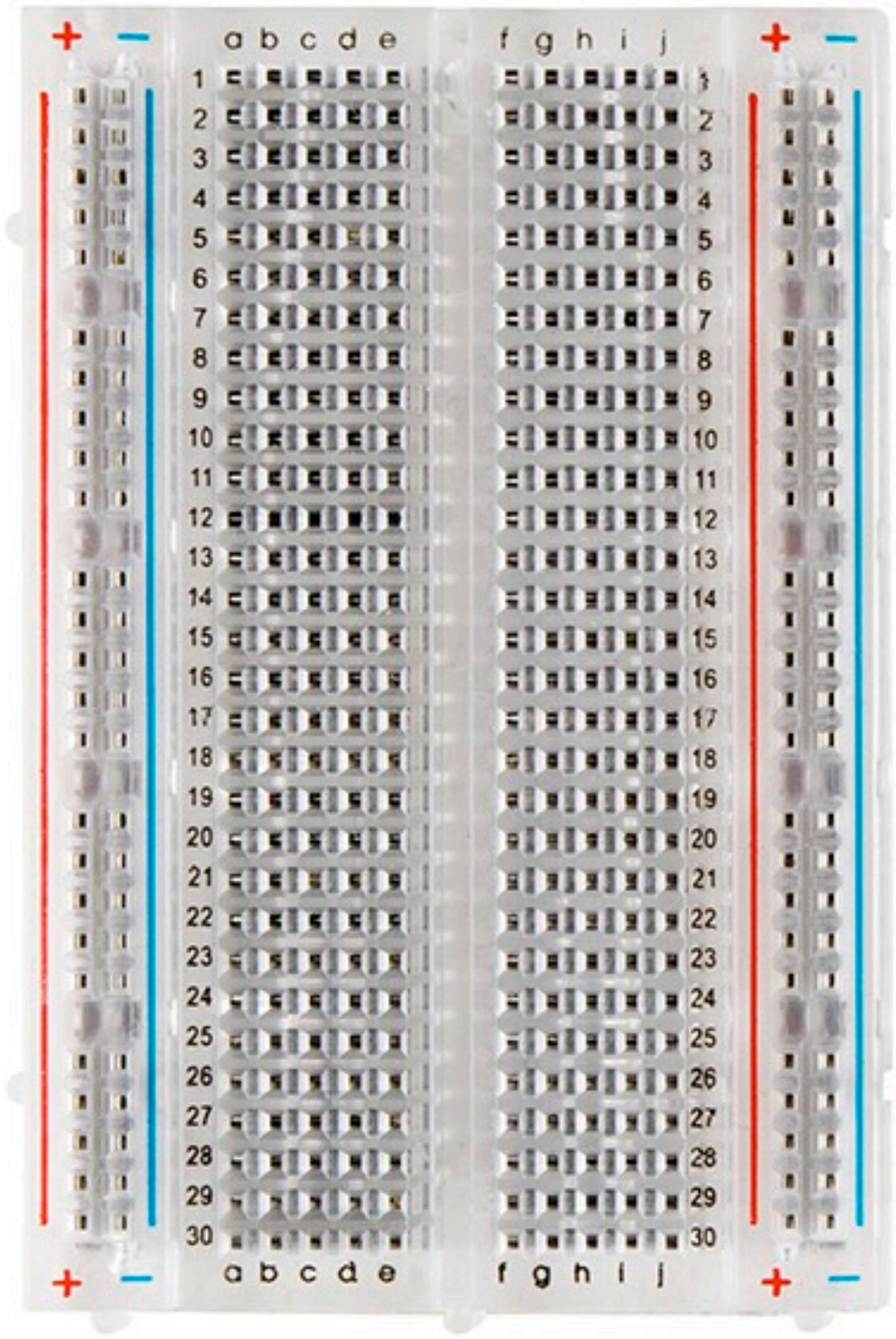
```
int  randNumber;

// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin LED_BUILTIN as an output.
  pinMode(LED_BUILTIN, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(LED_BUILTIN, HIGH);   // turn the LED on (HIGH is the voltage
  randNumber = random(50,500); //random number from 50 to 500
  delay(randNumber);                       // wait for random time
  digitalWrite(LED_BUILTIN, LOW);    // turn the LED off by making the voltag
  randNumber = random(50,500); //random number from 50 to 500
  delay(randNumber);                       // wait for random time
}
```

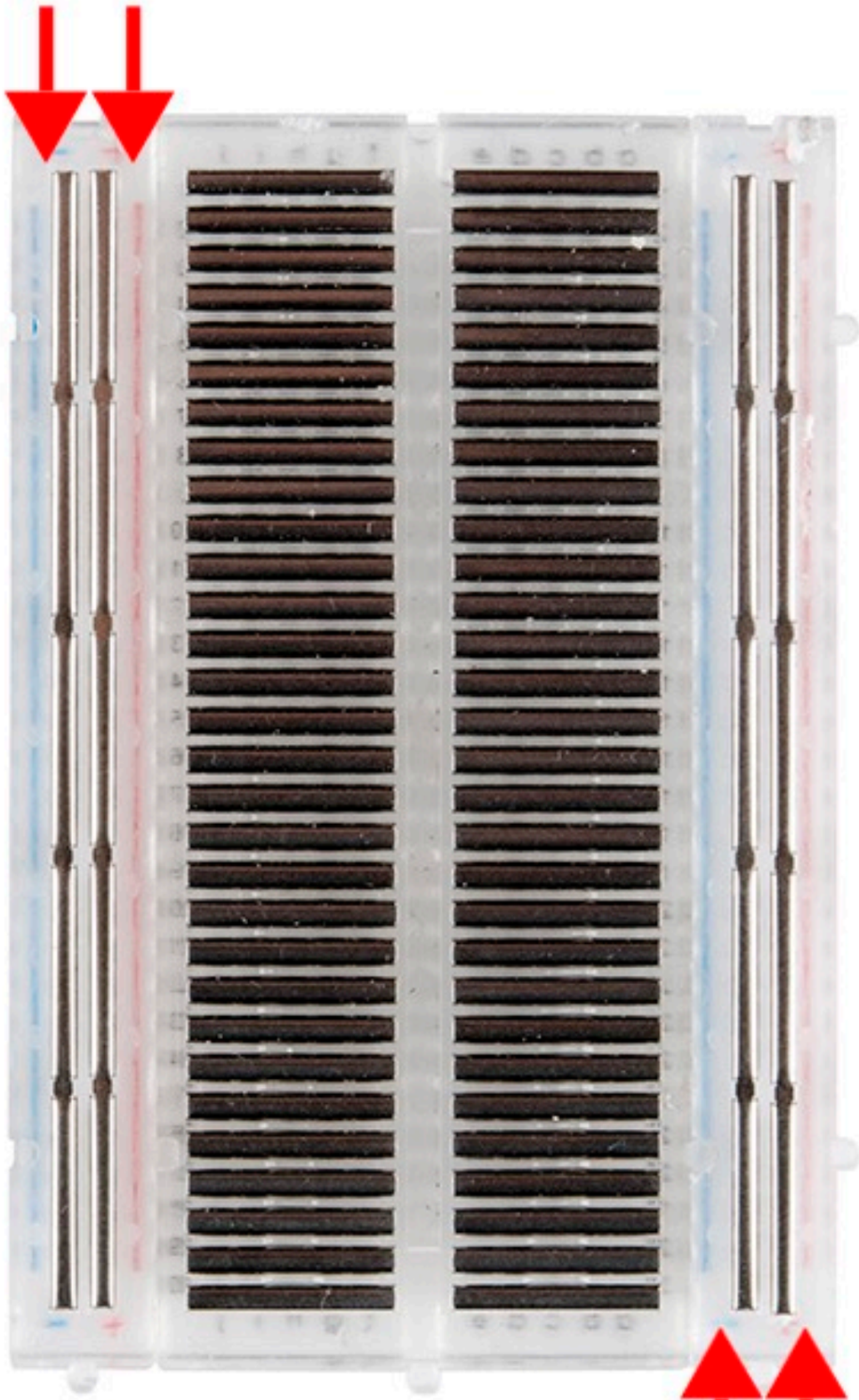
Done Saving.

Breadboard

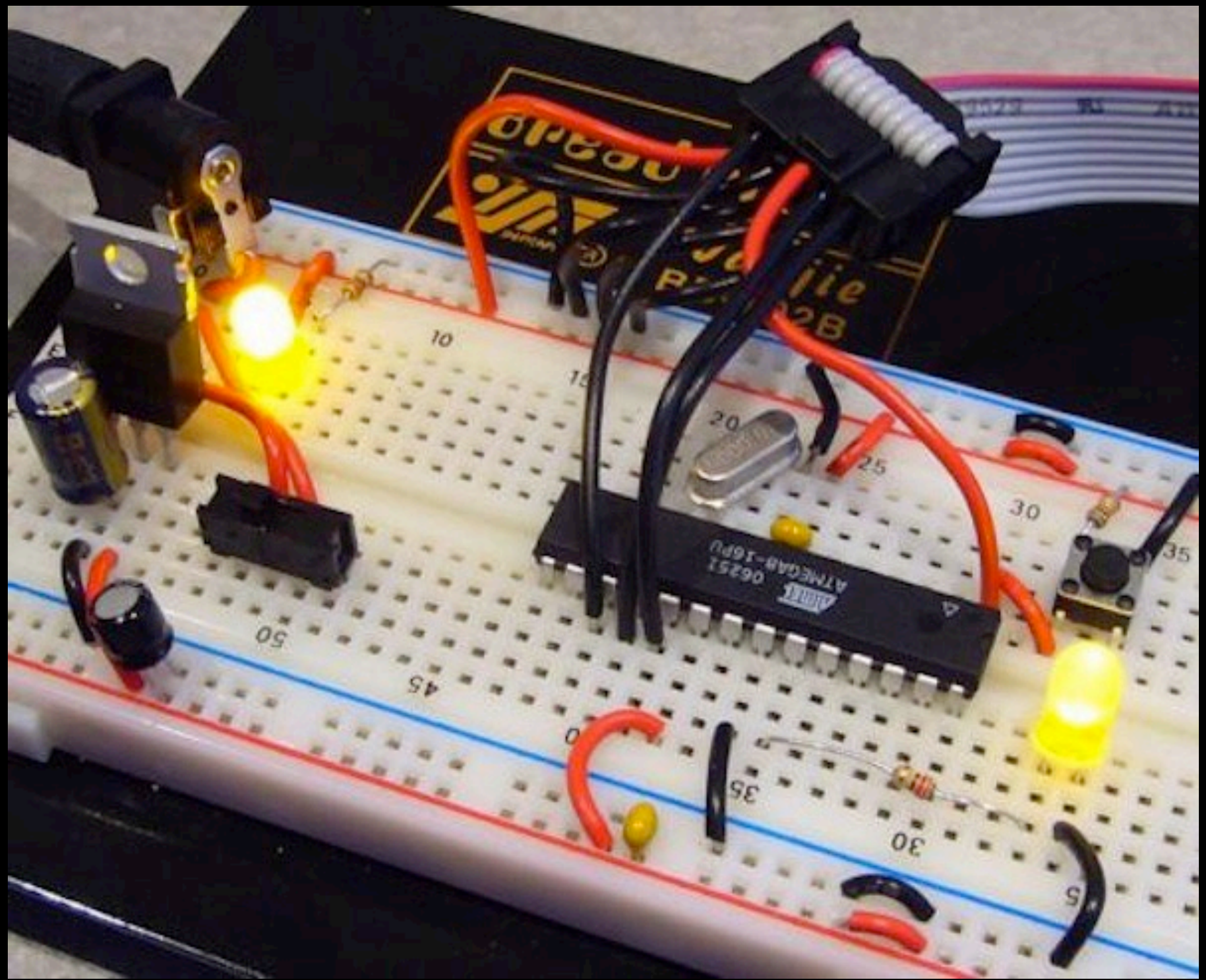


How to use a breadboard

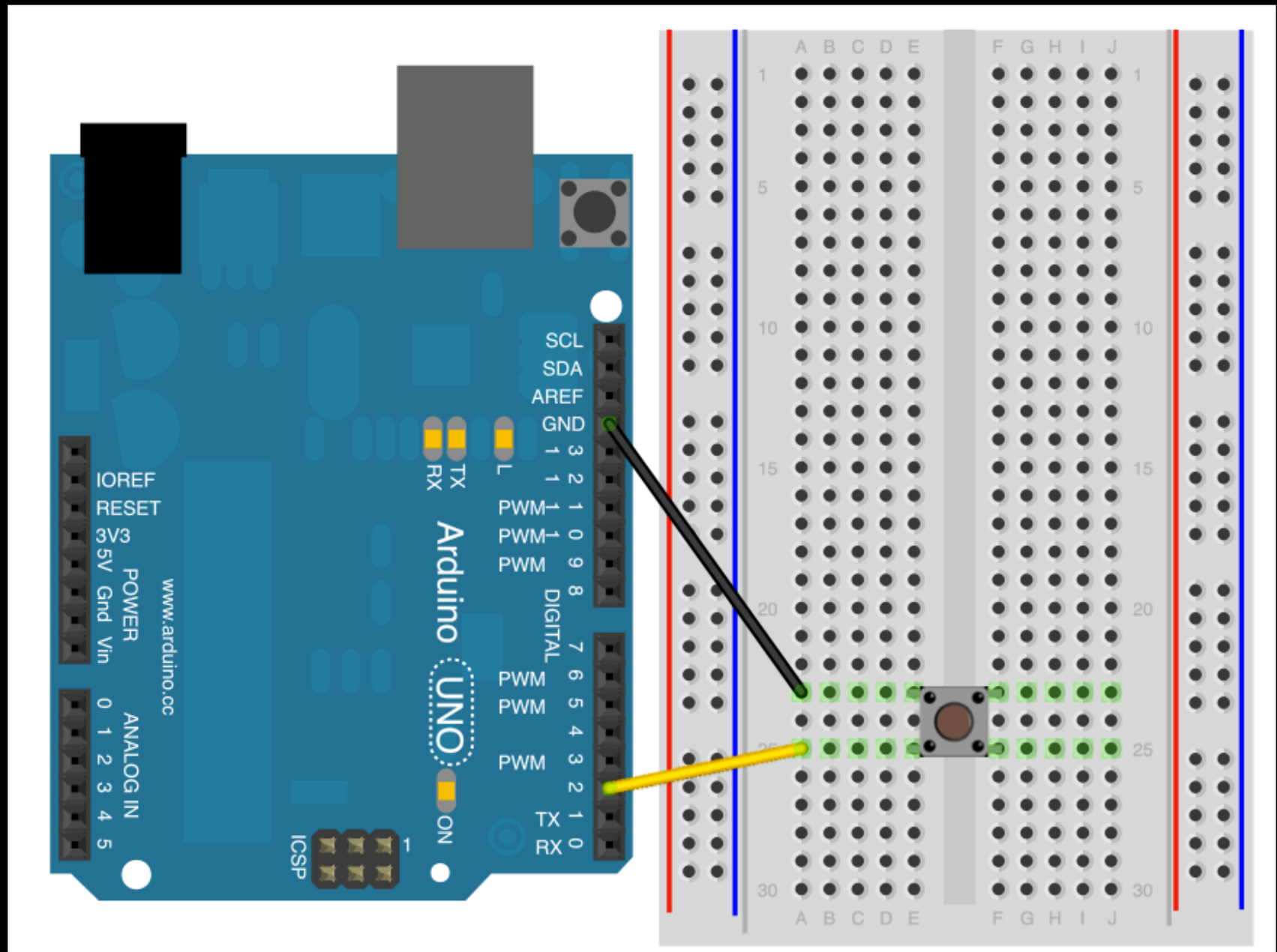




Circuit Example



Input Button



Input Button

Button | Arduino 1.8.1

Button

```
// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2;     // the number of the pushbutton pin
const int ledPin =  13;      // the number of the LED pin

// variables will change:
int buttonState = 0;         // variable for reading the pushbutton status

void setup() {
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
   //configure pin2 as an input and enable the internal pull-up resistor
  pinMode(buttonPin, INPUT_PULLUP);
}

void loop() {
  // read the state of the pushbutton value:
  buttonState = digitalRead(buttonPin);

  // check if the pushbutton is pressed.
  // if it is, the buttonState is HIGH:
  if (buttonState == HIGH) {
    // turn LED on:
    digitalWrite(ledPin, HIGH);
  } else {
    // turn LED off:
    digitalWrite(ledPin, LOW);
  }
}
```

Done Saving.

if / else

```
if (pinFiveInput < 500)
{
  // action A
}
else
{
  // action B
}
```

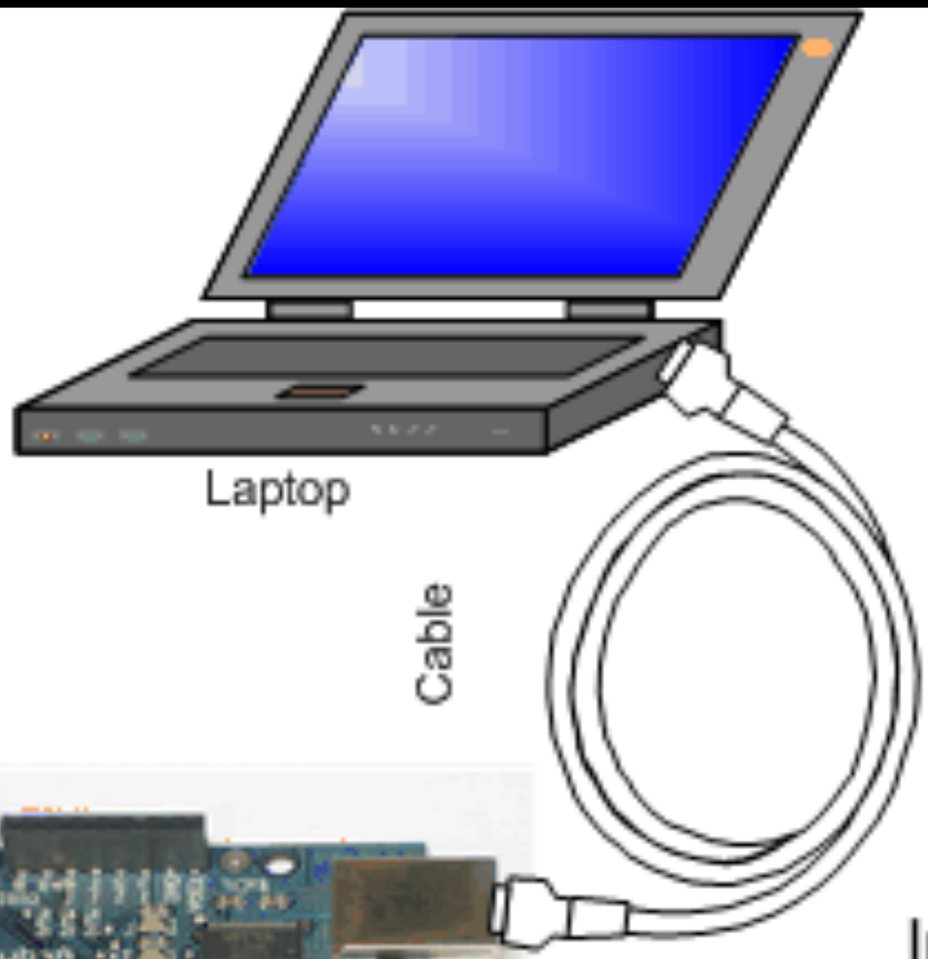
Switch Button

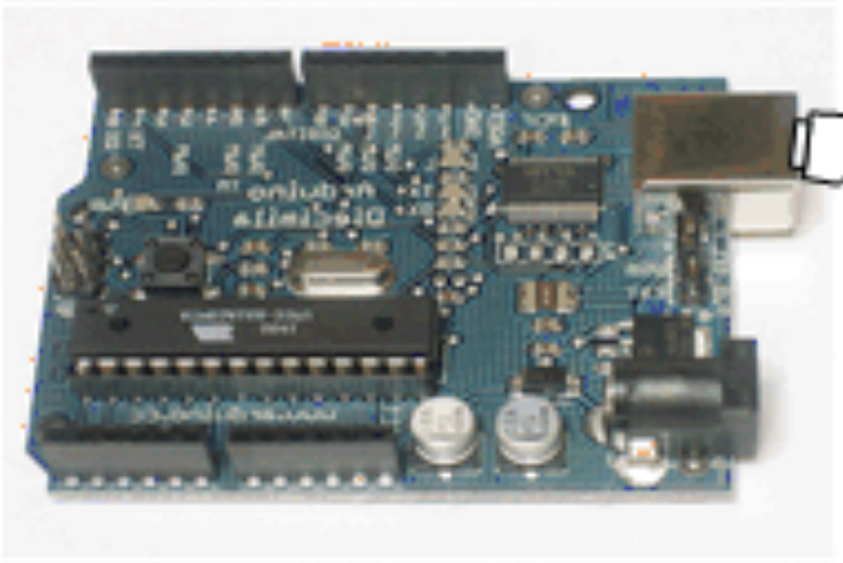
Debounce | Arduino 1.8.1

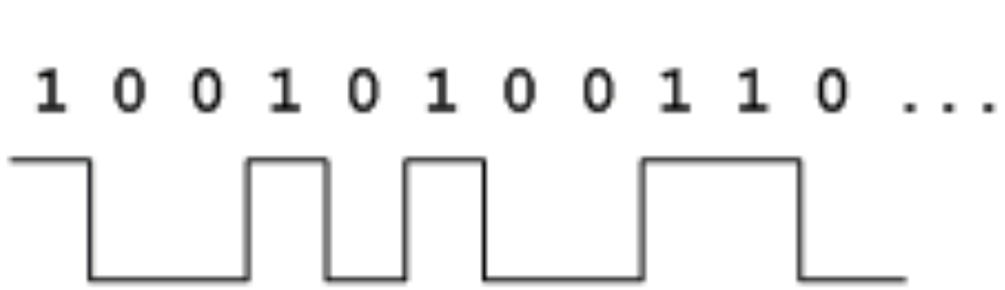
Debounce §

```
// constants won't change. They're used here to
// set pin numbers:
const int buttonPin = 2;    // the number of the pushbutton pin
const int ledPin = 13;      // the number of the LED pin

// Variables will change:
int ledState = HIGH;         // the current state of the output pin
int buttonState;             // the current reading from the input pin
int lastButtonState = LOW;   // the previous reading from the input pin

// the following variables are unsigned long's because the time, measured in miliseconds,
// will quickly become a bigger number than can be stored in an int.
unsigned long lastDebounceTime = 0;  // the last time the output pin was toggled
unsigned long debounceDelay = 50;    // the debounce time; increase if the output flickers

void setup() {
  pinMode(buttonPin, INPUT_PULLUP);
  pinMode(ledPin, OUTPUT);

  // set initial LED state
  digitalWrite(ledPin, ledState);
}

void loop() {
  // read the state of the switch into a local variable:
  int reading = digitalRead(buttonPin);

  // check to see if you just pressed the button
  // (i.e. the input went from LOW to HIGH),  and you've waited
  // long enough since the last press to ignore any noise:

  // If the switch changed, due to noise or pressing:
  if (reading != lastButtonState) {
    // reset the debouncing timer
    lastDebounceTime = millis();
  }

  if ((millis() - lastDebounceTime) > debounceDelay) {
    // whatever the reading is at, it's been there for longer
    // than the debounce delay, so take it as the actual current state:

    // if the button state has changed:
    if (reading != buttonState) {
      buttonState = reading;

      // only toggle the LED if the new button state is HIGH
      if (buttonState == HIGH) {
        ledState = !ledState;
      }
    }
  }

  // set the LED:
  digitalWrite(ledPin, ledState);

  // save the reading.  Next time through the loop,
  // it'll be the lastButtonState:
  lastButtonState = reading;
}
```

Serial Communication



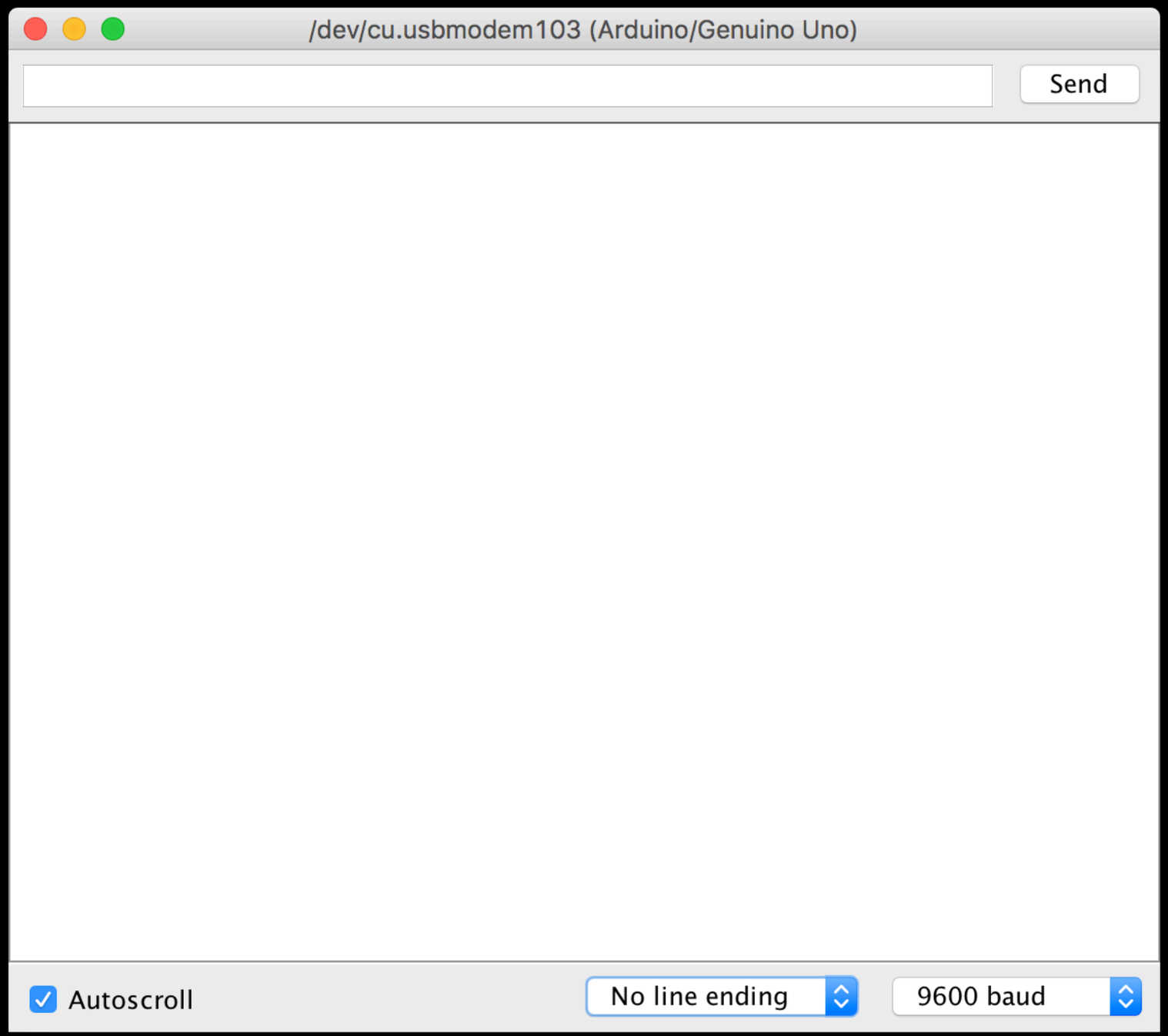




Information passes between the computer and Arduino through the USB cable. Information is transmitted as zeros (‘0’) and ones (‘1’)… also known as **bits**!

Serial Example

Serial Monitor



Input Serial

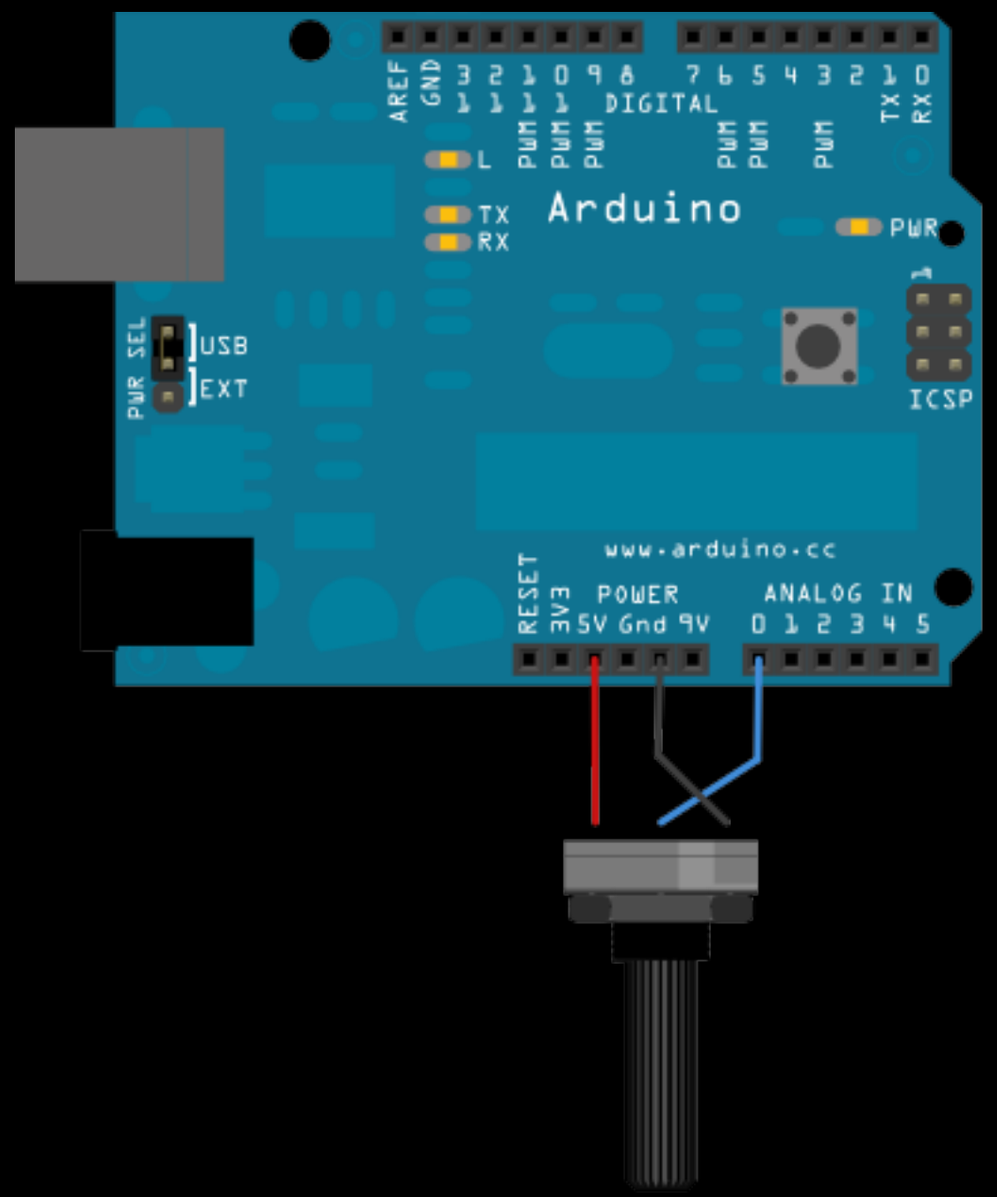
InputSerial | Arduino 1.8.1

InputSerial

```
1
2  void setup() {
3    //start serial connection
4    Serial.begin(9600);
5    //configure pin2 as an input and enable the internal pull-up resistor
6    pinMode(2, INPUT_PULLUP);
7    pinMode(13, OUTPUT);
8
9  }
10
11 void loop() {
12   //read the pushbutton value into a variable
13   int sensorVal = digitalRead(2);
14   //print out the value of the pushbutton
15   Serial.println(sensorVal);
16
17   // Keep in mind the pullup means the pushbutton's
18   // logic is inverted. It goes HIGH when it's open,
19   // and LOW when it's pressed. Turn on pin 13 when the
20   // button's pressed, and off when it's not:
21   if (sensorVal == HIGH) {
22     digitalWrite(13, LOW);
23   } else {
24     digitalWrite(13, HIGH);
25   }
26 }
```

Done Saving.

Potentiometer



Analog Read

AnalogRead | Arduino 1.8.1

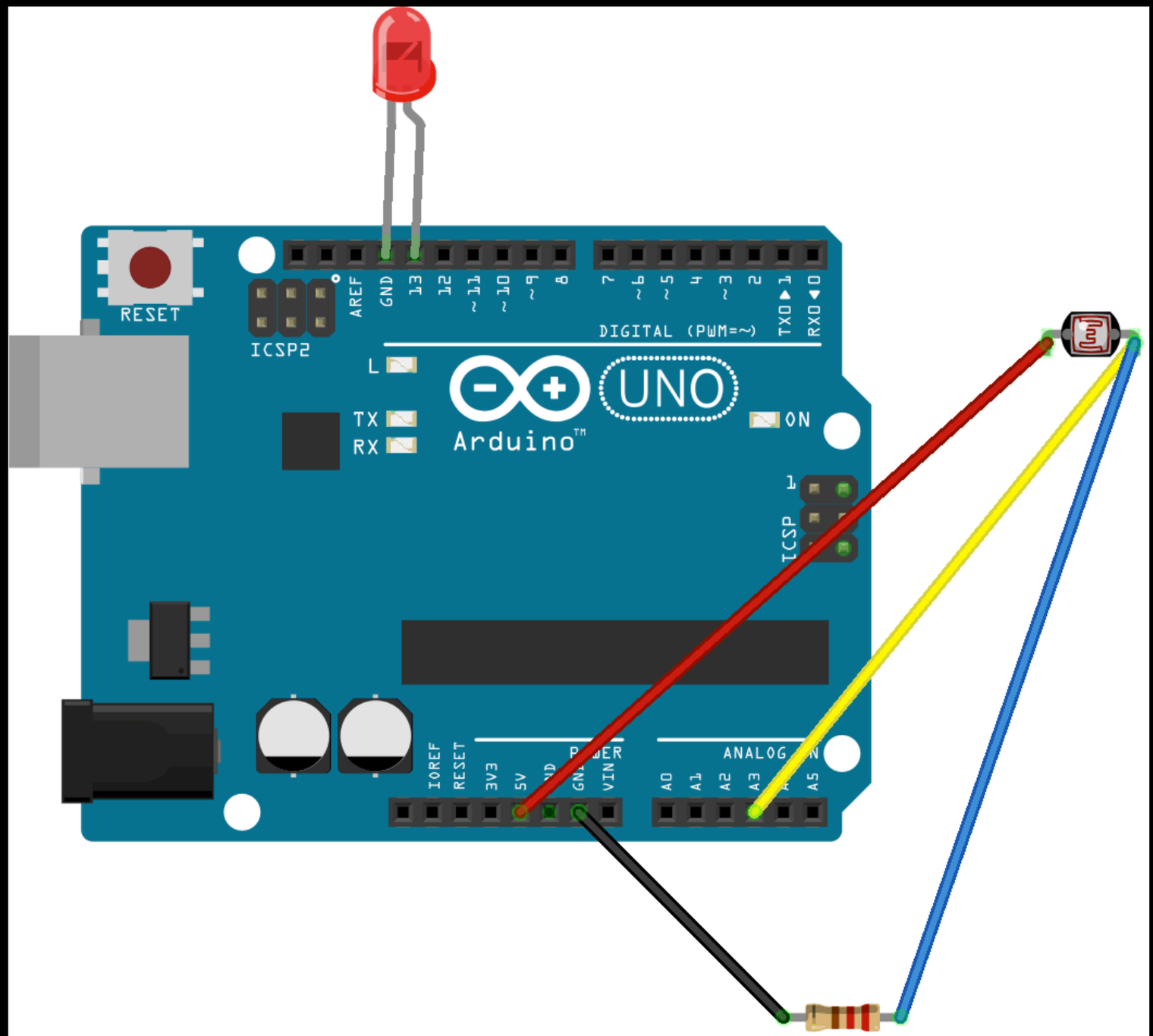
AnalogRead

```
1  int analogPin = 0;      // potentiometer wiper (middle terminal) connected to analog pin 0
2                          // outside leads to ground and +5V
3  int val = 0;            // variable to store the value read
4
5  void setup()
6  {
7    Serial.begin(9600);          //  setup serial
8  }
9
10 void loop()
11 {
12   val = analogRead(analogPin);    // read the input pin
13   Serial.println(val);             // debug value
14 }
15
16
```

Done Saving.

Components

Photoresistor



Photoresistor

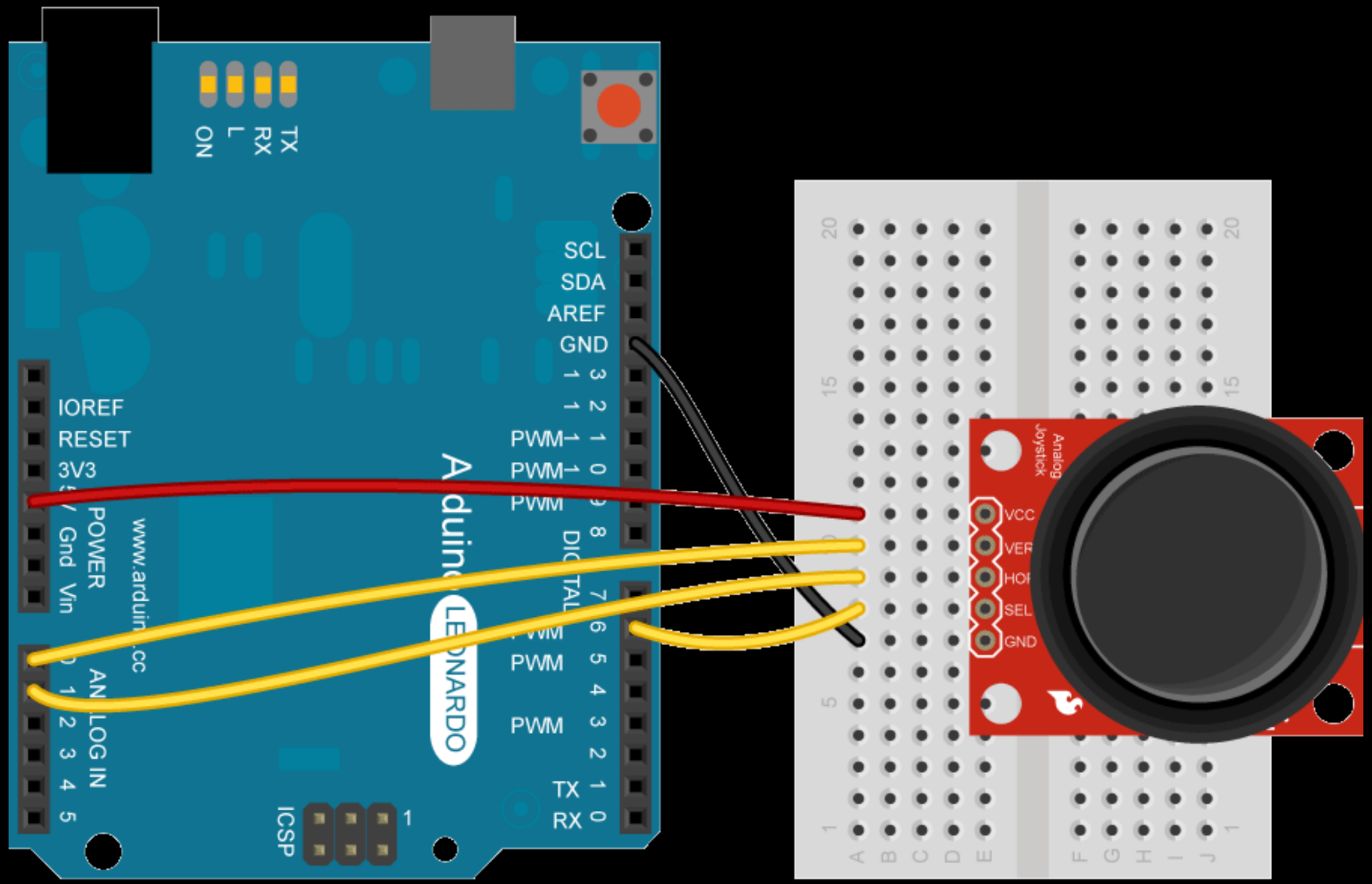
PhotoResistorExample | Arduino 1.8.1

PhotoResistorExample

```
1  void setup() {
2      pinMode(13, OUTPUT);
3  }
4
5  void loop() {
6      int lightLevel = analogRead(3);
7      delay(100);
8      if (lightLevel < 20) { //20 is an arbitrary threshold to turn on LED
9          digitalWrite(13, HIGH);
10     }
11     else {
12         digitalWrite(13, LOW);
13     }
14 }
15
16
```

Done Saving.

Joystick


ON
L
TX
RX
SCL
SDA
AREF
GND
IOREF
RESET
3V3
POWER
Gnd
Vin
ANALOG IN
PWM
DIGITAL
TX
RX
ICSP
LEONARDO
www.arduino.cc
Analog Joystick
VCC
VER
HOR
SEL
GND

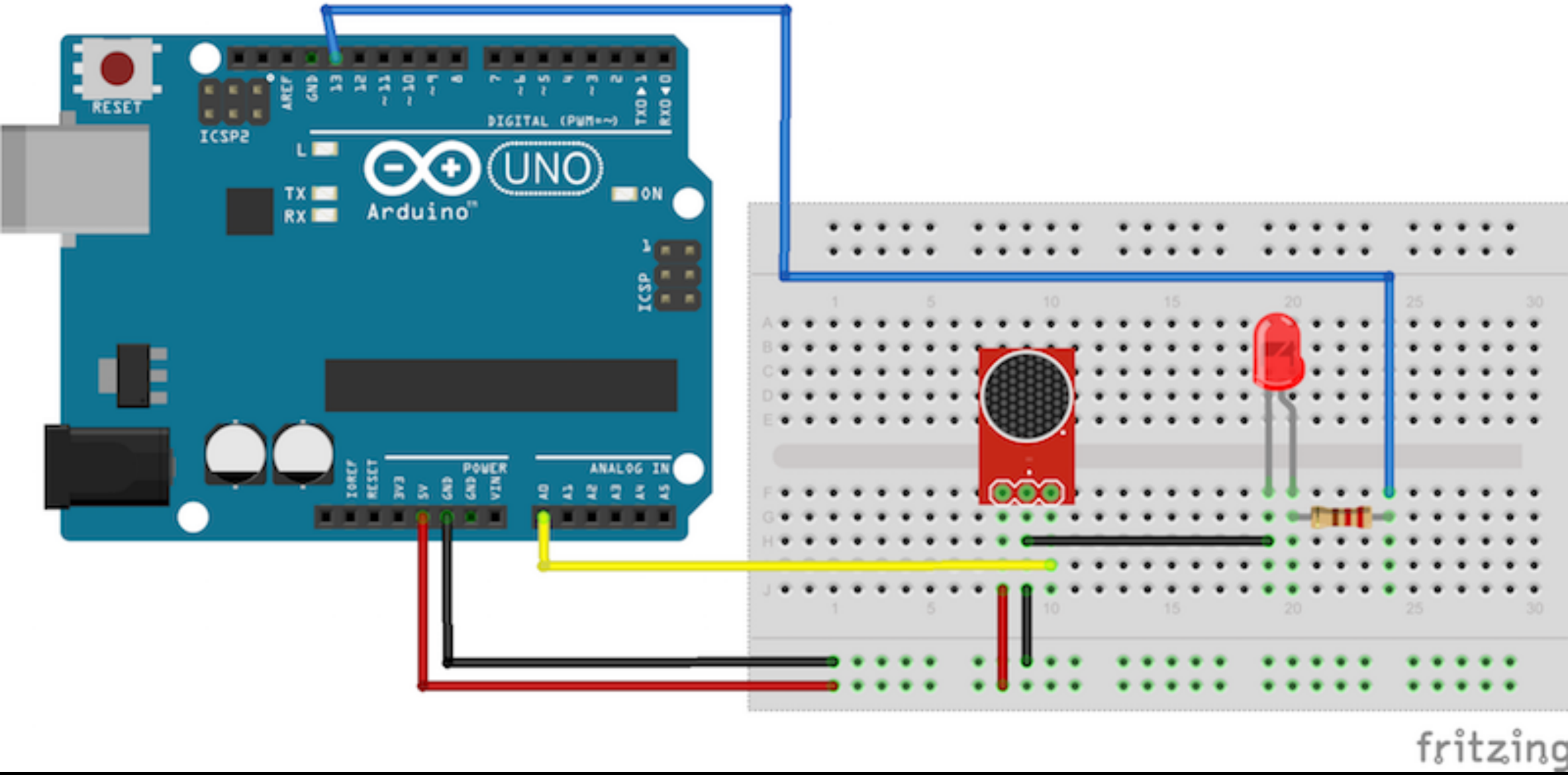
Joystick

Joystick | Arduino 1.8.1

Joystick

```
int ledPin = 13;
int joyPin1 = 0;                 // slider variable connecetd to analog pin 0
int joyPin2 = 1;                 // slider variable connecetd to analog pin 1
int value1 = 0;                  // variable to read the value from the analog pin 0
int value2 = 0;                  // variable to read the value from the analog pin 1

void setup() {
 pinMode(ledPin, OUTPUT);              // initializes digital pins 0 to 7 as outputs
 Serial.begin(9600);
}

int treatValue(int data) {
 return (data * 9 / 1024) + 48;
}

void loop() {
 // reads the value of the variable resistor
 value1 = analogRead(joyPin1);
 // this small pause is needed between reading
 // analog pins, otherwise we get the same value twice
 delay(100);
 // reads the value of the variable resistor
 value2 = analogRead(joyPin2);

 digitalWrite(ledPin, HIGH);
 delay(value1);
 digitalWrite(ledPin, LOW);
 delay(value2);
 Serial.print('J');
 Serial.print(treatValue(value1));
 Serial.println(treatValue(value2));
}
```

Sound Sensor



Sound Sensor

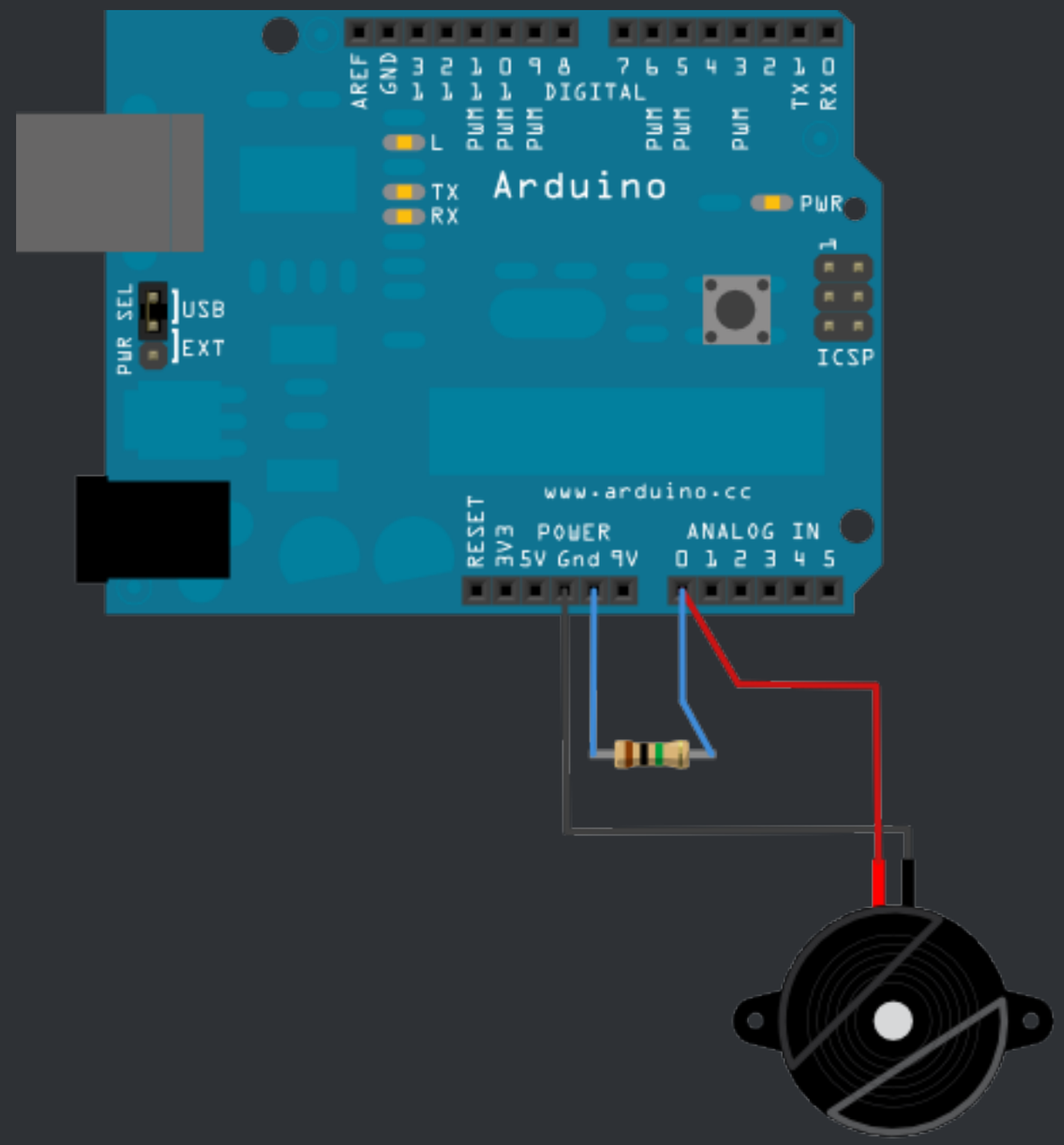
SoundSensorExample | Arduino 1.8.1

SoundSensorExample

```
/*Sound Sensor Example*/
int led = 13;
int threshold = 500; //Change this value if you want change the sensibility of sensor
int sound;
int led_value;

void setup()
{
    pinMode(led, OUTPUT);
    digitalWrite(led,LOW);
    led_value=0;
}

void loop()
{
    sound = analogRead(A0); // Reads the value from the Analog PIN A0
    if(sound>=threshold){
        led_value=!led_value;}
    digitalWrite(led, led_value);
    delay(100);
}

```

Done Saving.

Piezo Element



Piezo Element

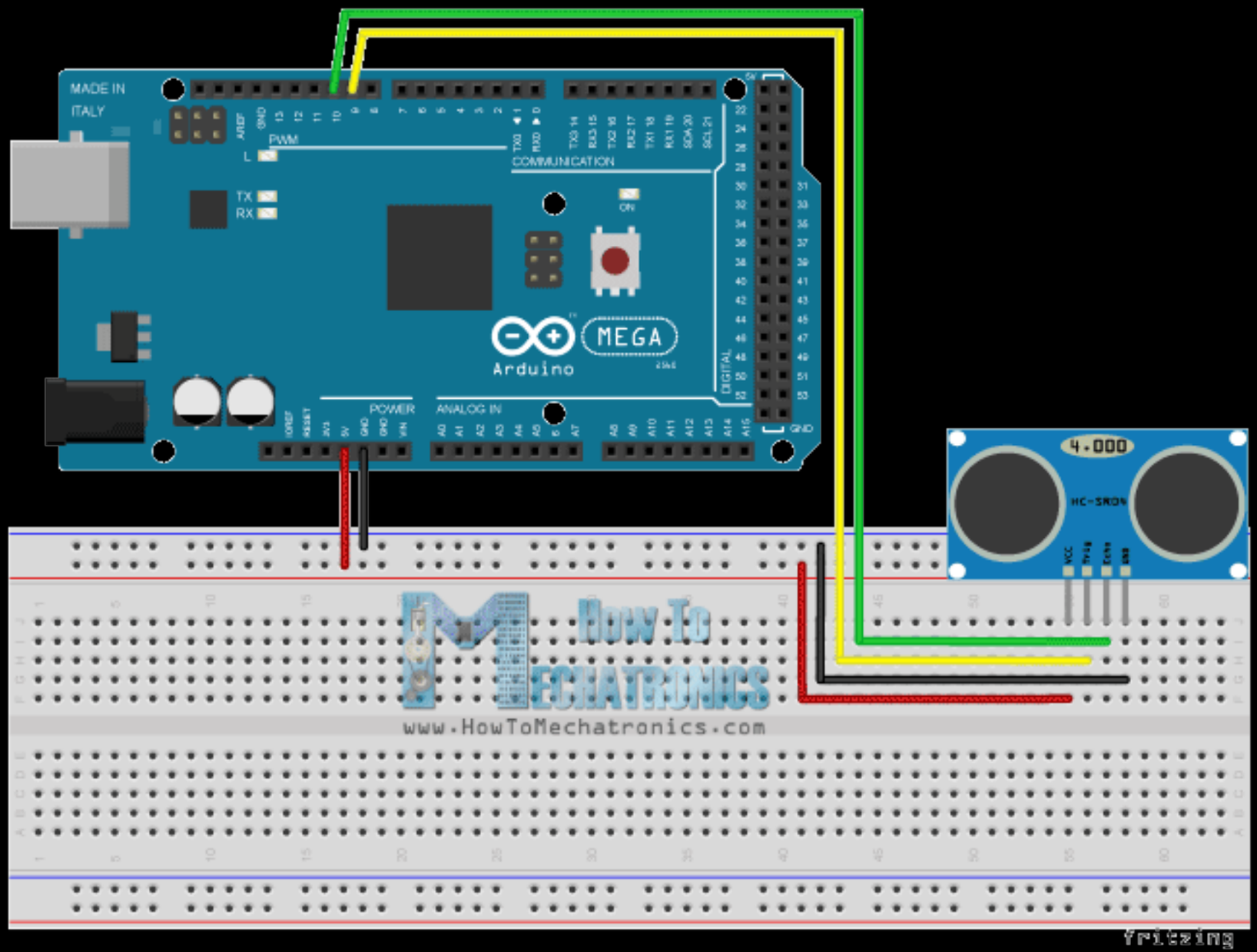
Piezo | Arduino 1.8.1

Piezo

```
1
2  // these constants won't change:
3  const int ledPin = 13;      // led connected to digital pin 13
4  const int knockSensor = A0; // the piezo is connected to analog pin 0
5  const int threshold = 100;  // threshold value to decide when the detected sound is a knock or not
6
7
8  // these variables will change:
9  int sensorReading = 0;      // variable to store the value read from the sensor pin
10 int ledState = LOW;         // variable used to store the last LED status, to toggle the light
11
12 void setup() {
13   pinMode(ledPin, OUTPUT); // declare the ledPin as as OUTPUT
14   Serial.begin(9600);       // use the serial port
15 }
16
17 void loop() {
18   // read the sensor and store it in the variable sensorReading:
19   sensorReading = analogRead(knockSensor);
20
21   // if the sensor reading is greater than the threshold:
22   if (sensorReading >= threshold) {
23     // toggle the status of the ledPin:
24     ledState = !ledState;
25     // update the LED pin itself:
26     digitalWrite(ledPin, ledState);
27     // send the string "Knock!" back to the computer, followed by newline
28     Serial.println("Knock!");
29   }
30   delay(100);  // delay to avoid overloading the serial port buffer
31 }
32
33
```

Done Saving.

Distance Sensor



Distance Sensor

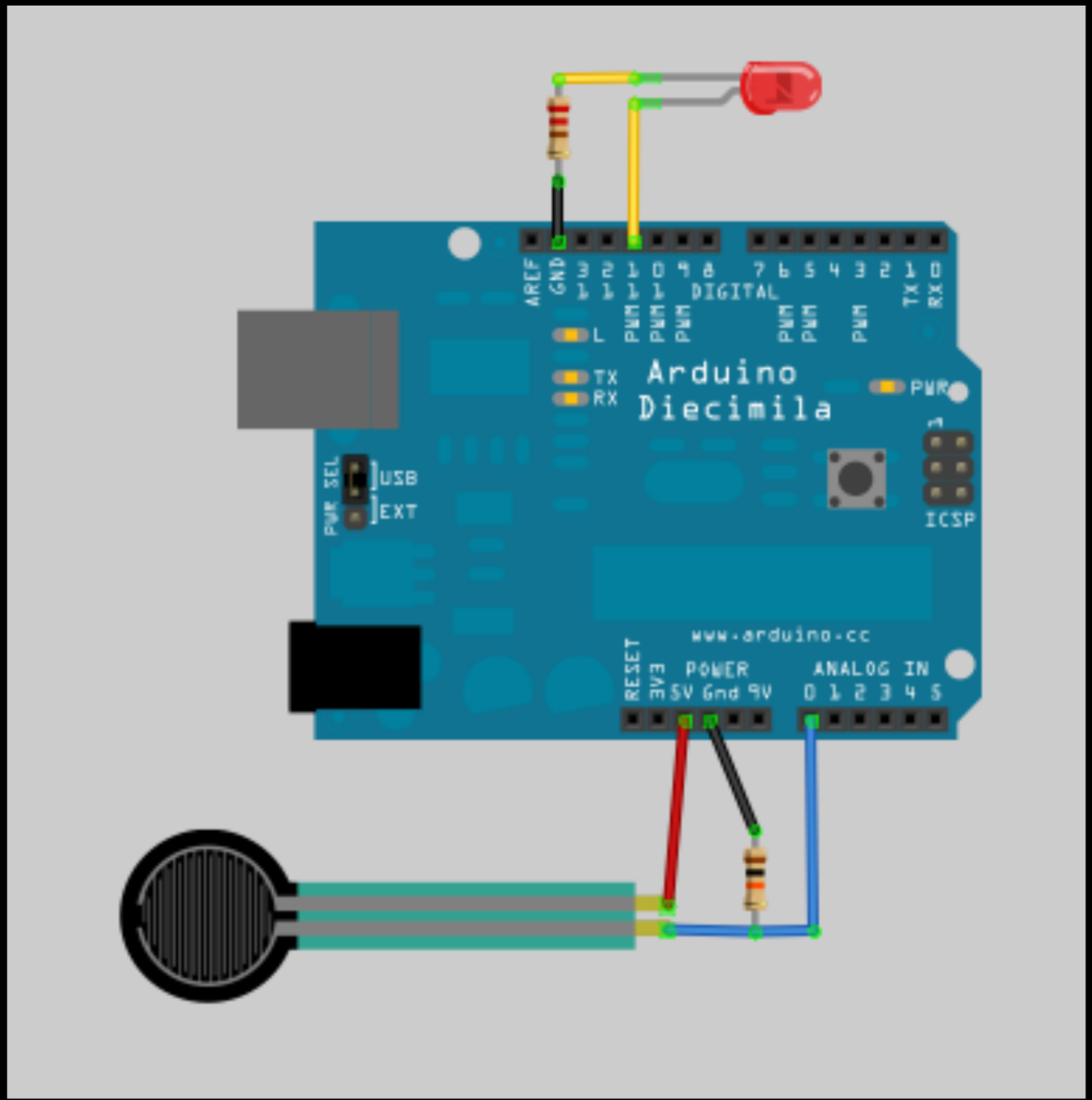
DistanceSensor | Arduino 1.8.1

DistanceSensor

```
/*
* Ultrasonic Sensor HC-SR04 and Arduino Tutorial
*
* Crated by Dejan Nedelkovski,
* www.HowToMechatronics.com
*
*/
// defines pins numbers
const int trigPin = 9;
const int echoPin = 10;
// defines variables
long duration;
int distance;
void setup() {
pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
pinMode(echoPin, INPUT); // Sets the echoPin as an Input
Serial.begin(9600); // Starts the serial communication
}
void loop() {
// Clears the trigPin
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance= duration*0.034/2;
// Prints the distance on the Serial Monitor
Serial.print("Distance: ");
Serial.println(distance);
}
```

Done Saving.

Force Sensing Resistor



Force Sensing Resistor

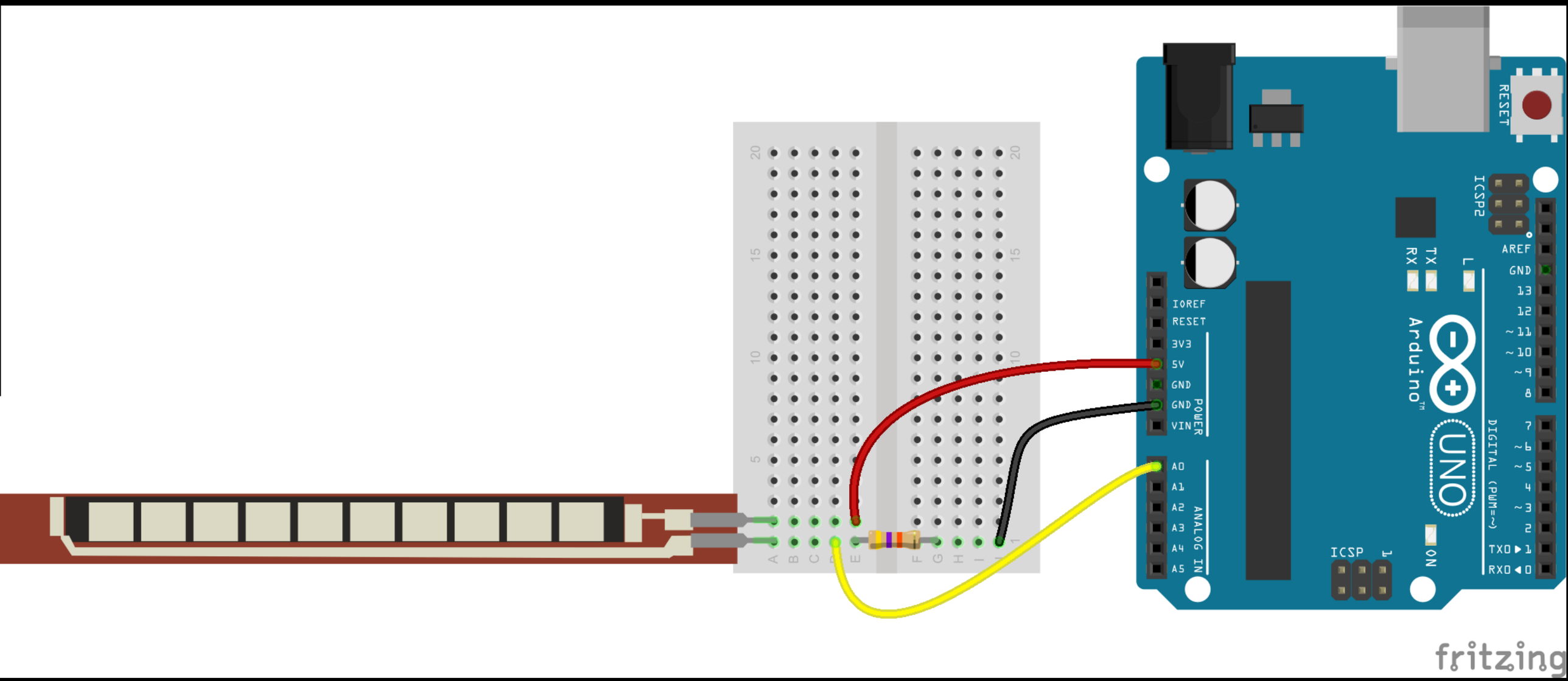
ForceSensitiveResistor | Arduino 1.8.1

ForceSensitiveResistor

```
1
2  int fsrAnalogPin = 0; // FSR is connected to analog 0
3  int LEDpin = 11;      // connect Red LED to pin 11 (PWM pin)
4  int fsrReading;      // the analog reading from the FSR resistor divider
5  int LEDbrightness;
6
7  void setup(void) {
8    Serial.begin(9600);   // We'll send debugging information via the Serial monitor
9    pinMode(LEDpin, OUTPUT);
10 }
11
12 void loop(void) {
13   fsrReading = analogRead(fsrAnalogPin);
14   Serial.print("Analog reading = ");
15   Serial.println(fsrReading);
16
17   // we'll need to change the range from the analog reading (0-1023) down to the range
18   // used by analogWrite (0-255) with map!
19   LEDbrightness = map(fsrReading, 0, 1023, 0, 255);
20   // LED gets brighter the harder you press
21   analogWrite(LEDpin, LEDbrightness);
22
23   delay(100);
24 }
25
26
```

Done Saving.

Flex Sensor


RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
TX
RX
L
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
Arduino™
UNO
DIGITAL (PWM=~)
7
~6
~5
4
~3
2
TX0 1
RX0 0
ON
ICSP
1
20
15
10
5
A
B
C
D
E
F
G
H
I
J
fritzing

Flex Sensor

FlexSensor | Arduino 1.8.1

FlexSensor

```
1
2  const int FLEX_PIN = A0; // Pin connected to voltage divider output
3
4  // Measure the voltage at 5V and the actual resistance of your
5  // 47k resistor, and enter them below:
6  const float VCC = 4.98; // Measured voltage of Ardunio 5V line
7  const float R_DIV = 47500.0; // Measured resistance of 3.3k resistor
8
9  // Upload the code, then try to adjust these values to more
10 // accurately calculate bend degree.
11 const float STRAIGHT_RESISTANCE = 37300.0; // resistance when straight
12 const float BEND_RESISTANCE = 90000.0; // resistance at 90 deg
13
14 void setup()
15 {
16   Serial.begin(9600);
17   pinMode(FLEX_PIN, INPUT);
18 }
19
20 void loop()
21 {
22   // Read the ADC, and calculate voltage and resistance from it
23   int flexADC = analogRead(FLEX_PIN);
24   float flexV = flexADC * VCC / 1023.0;
25   float flexR = R_DIV * (VCC / flexV - 1.0);
26   Serial.println("Resistance: " + String(flexR) + " ohms");
27
28   // Use the calculated resistance to estimate the sensor's
29   // bend angle:
30   float angle = map(flexR, STRAIGHT_RESISTANCE, BEND_RESISTANCE,
31                     0, 90.0);
32   Serial.println("Bend: " + String(angle) + " degrees");
33   Serial.println();
34
35   delay(500);
36 }
37
```

Done Saving.

Questions?

Imanol Gómez
imanolgomez.net
yo@imanolgomez.net