WELCOME TO THE

# CREATIVE ROBOTICS CLUB

# WHAT DO WE DO AT THE **CREATIVE ROBOTICS CLUB?**

We learn how to use electricity,
robotics and code to make things

We make art, design, or social robotics
- we support all disciplines

We reuse and repurpose where we can

We have fun

# WE ARE OPEN TO YOUR FEEDBACK!

Are there things you want us to talk about?

A different way of running you think will work?

Skills you want to share?

We are a club for students, and we welcome your suggestions and input

Welcome to the

# CREATIVE ROBOTICS CLUB

Want to talk?

Need help?

**Join us on Discord**

# BUT FIRST LETS TALK ABOUT...

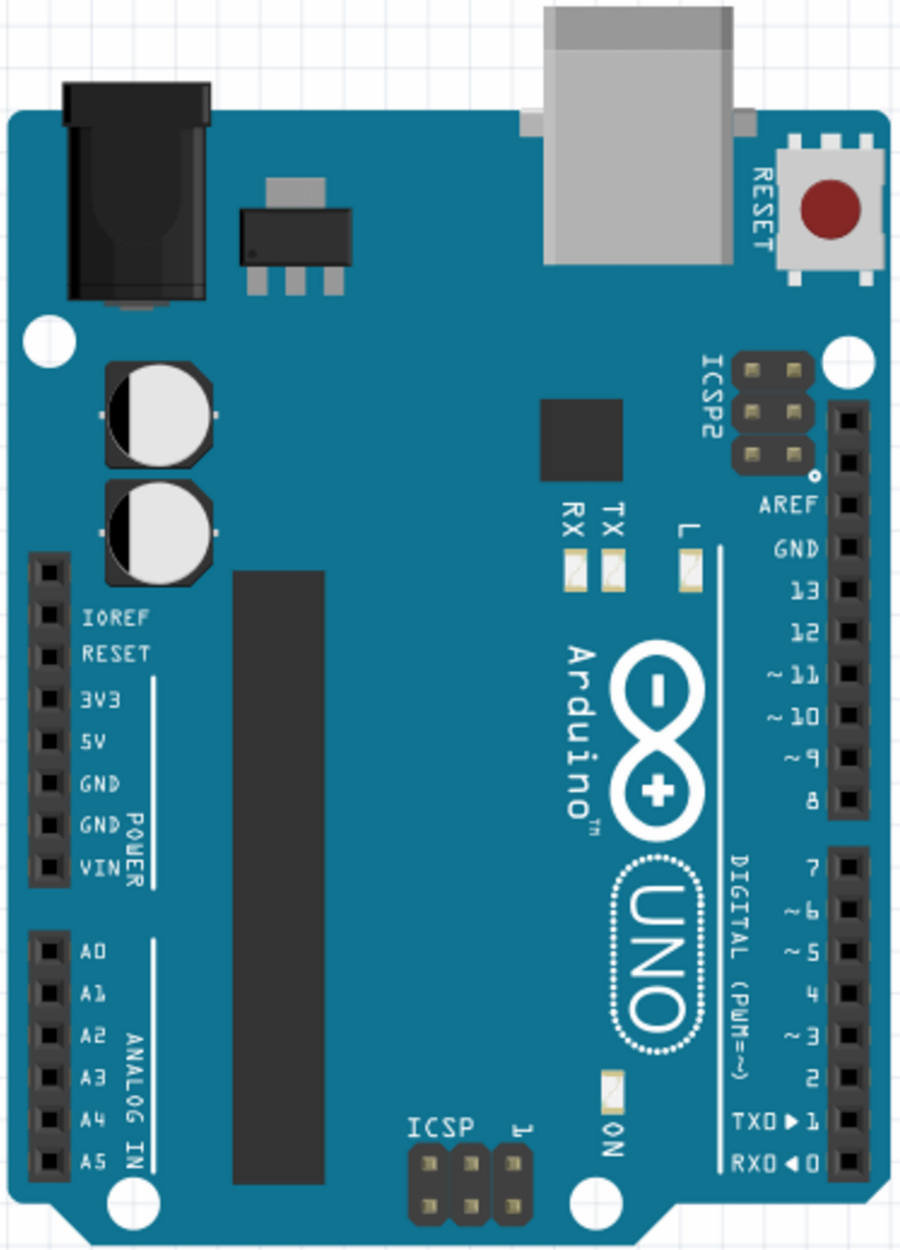
# LAST WEEK

# ARDUINO

This is an Arduino Uno

The easiest way to think about it is as a box of dimmers and switches

It can also read in information. We can use that information to drive things with electricity, we'll talk more about that next week




Coding is like
cooking

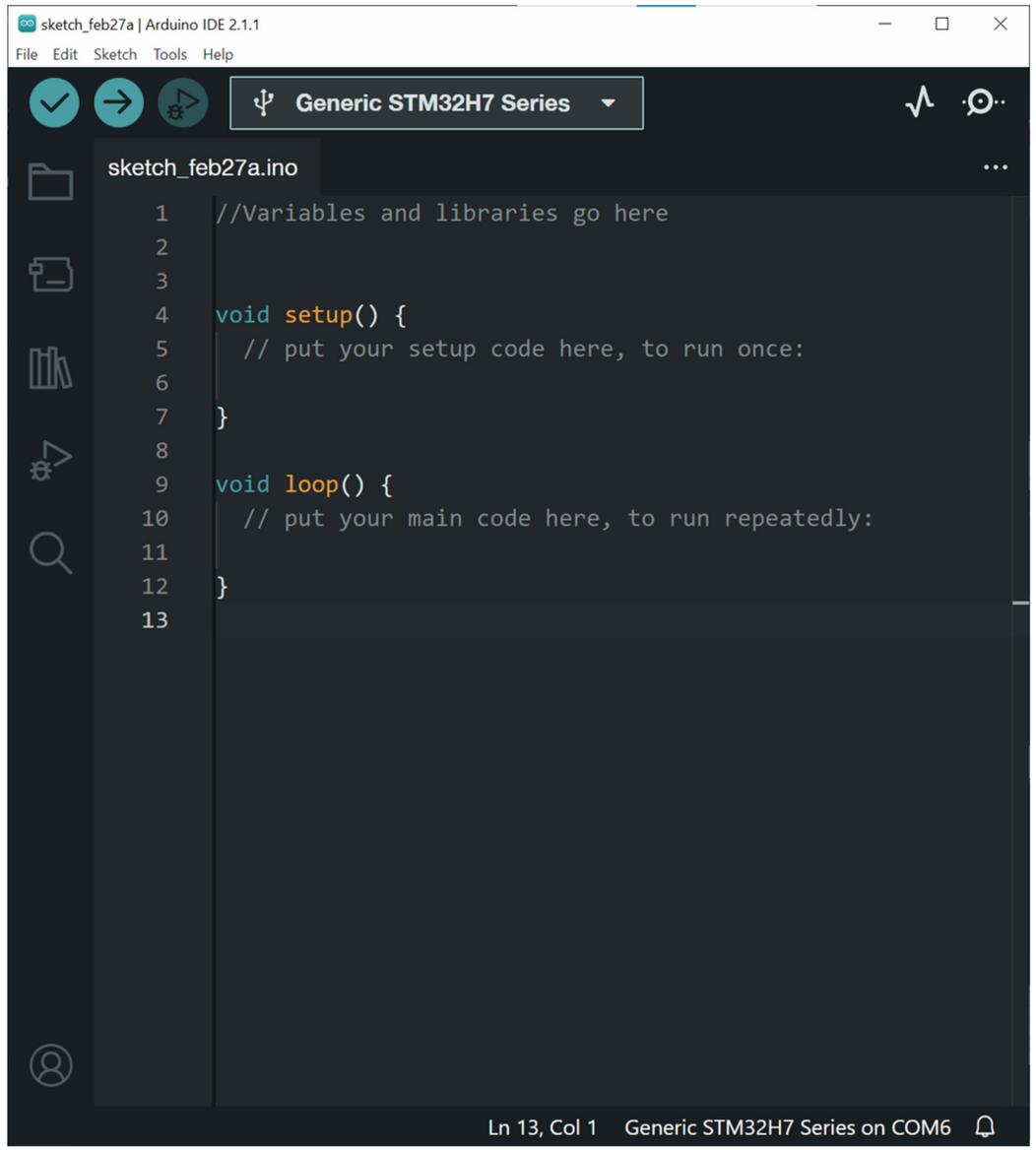
# THINK OF IT LIKE A RECIPE

At the top we tell the program what ingredients we need.

We call this declaring our variables.

We also add in whatever libraries we are using, e.g. #include <Servo.h>

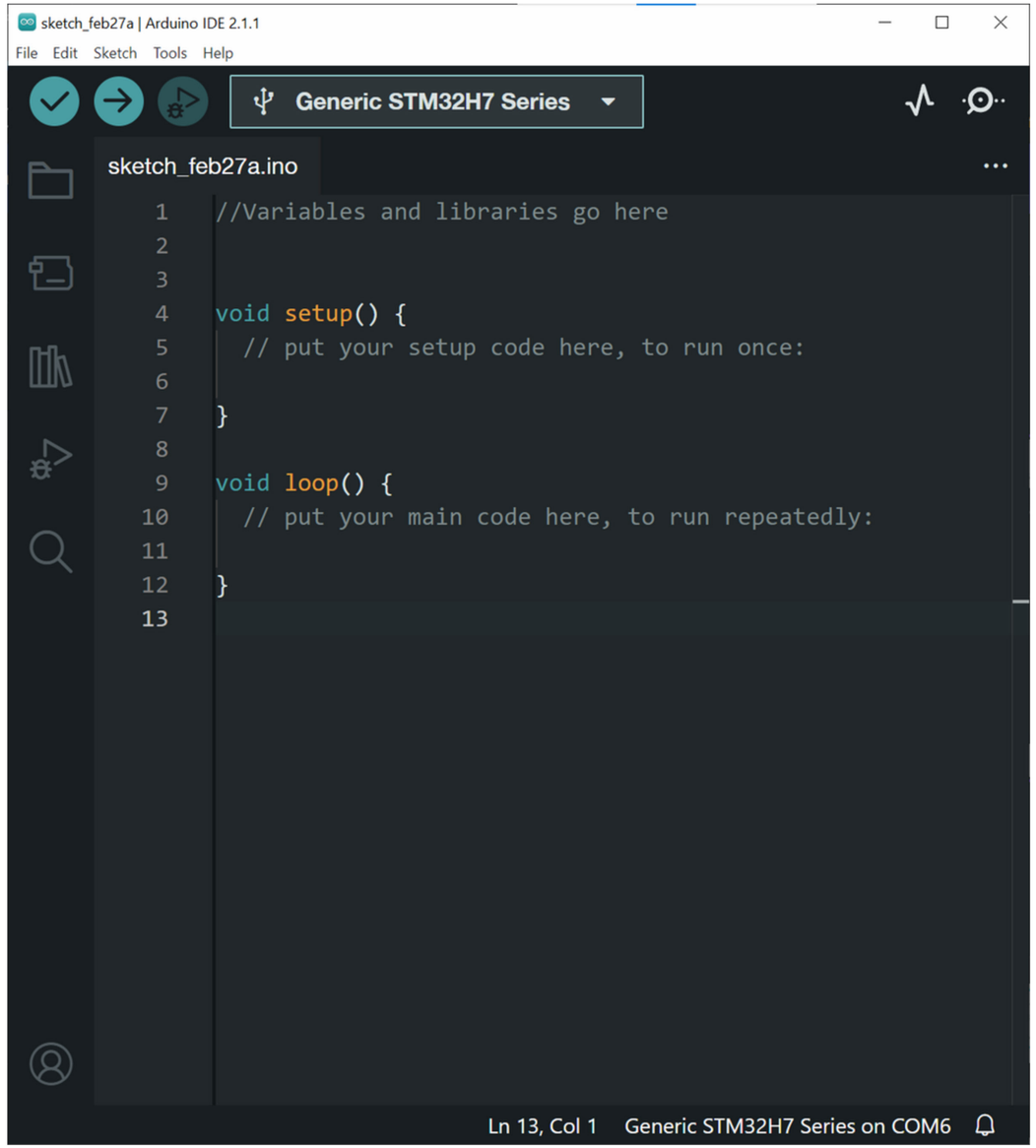
We declare our variables once.



# THINK OF IT LIKE A RECIPE

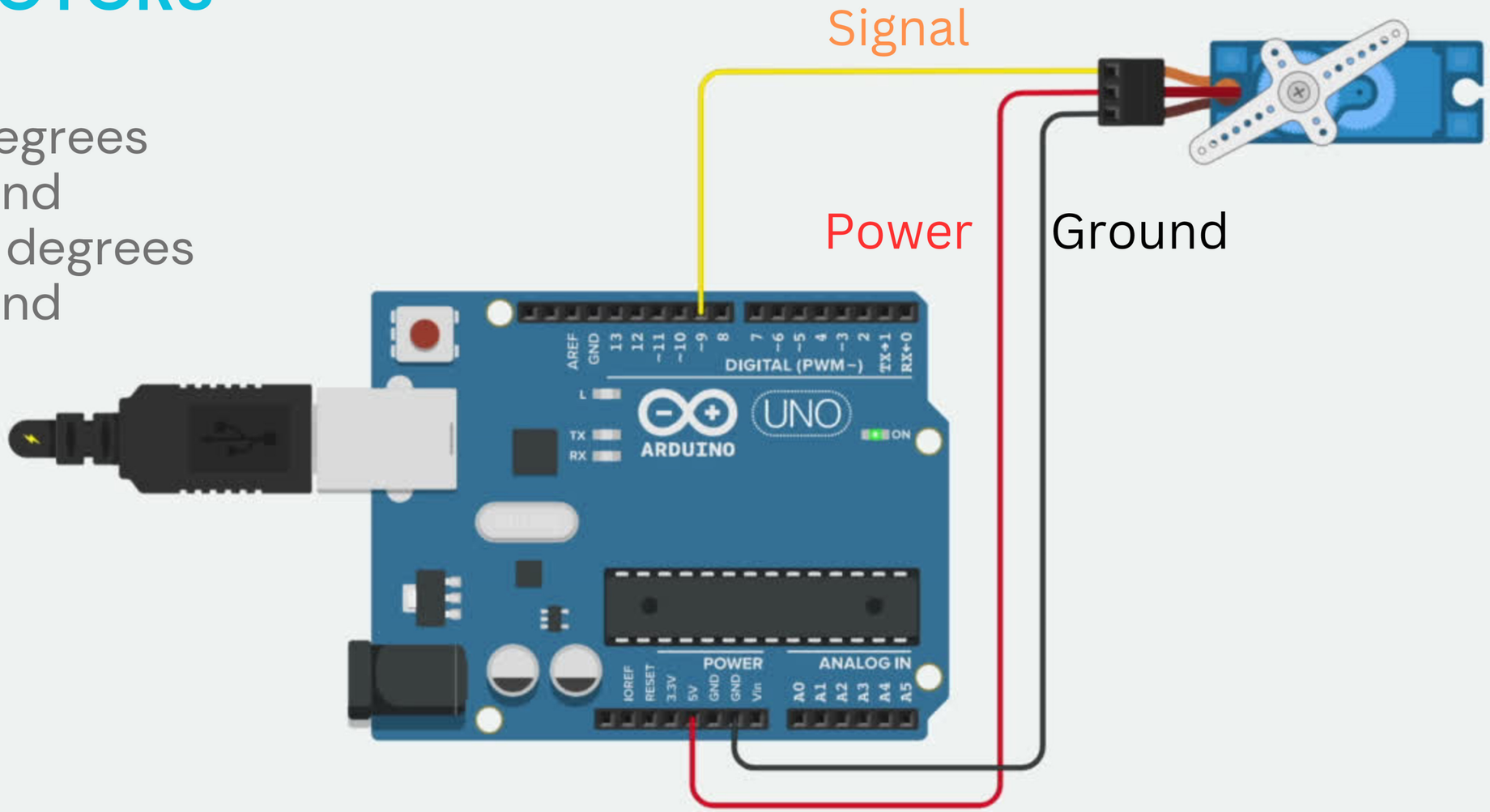
void setup() is like preparing our ingredients. It also only runs at the start of our program – when the board powers on. We use this to setup our Arduino for use, e.g. pinMode(#, INPUT / OUTPUT)

void loop() will run after void setup(), looping over and over again while the board is powered on.

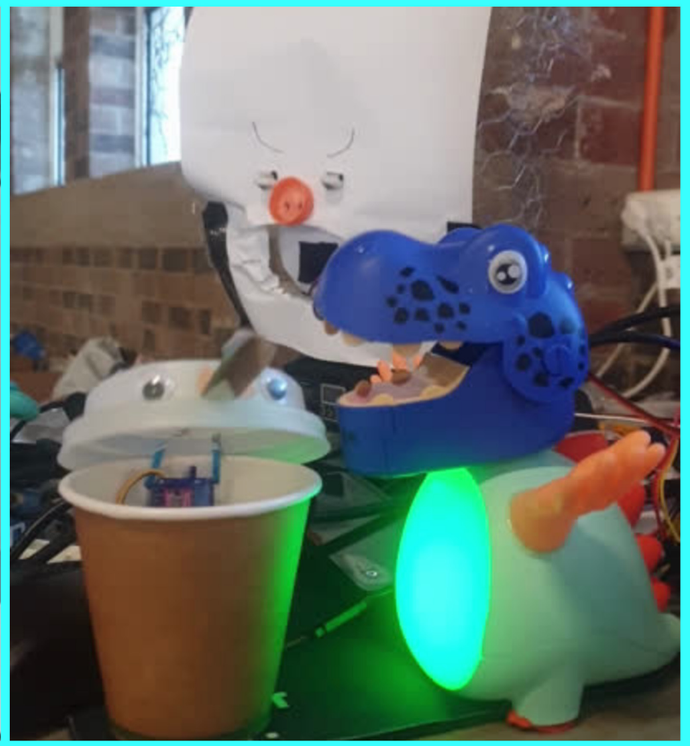


# SERVO MOTORS

Servo will:
Move to 0 degrees
Pause 1 second
Move to 180 degrees
Pause 1 second
Repeat



# WHAT ARE WE DOING TODAY AT THE CREATIVE ROBOTICS CLUB?



## SENSORS + CONTROL

# ARDUINO

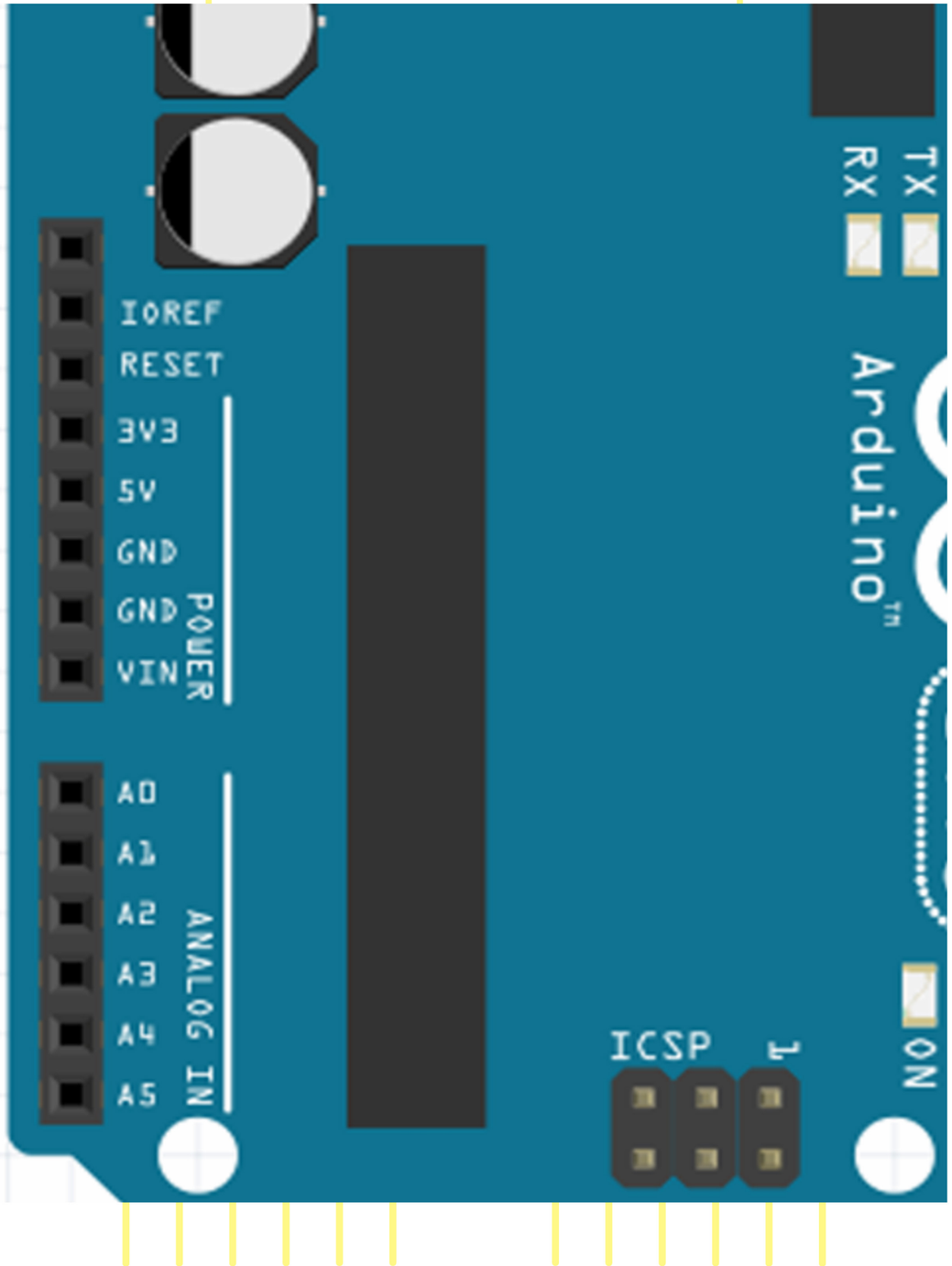
Analog In pins are located in the bottom left

This section can be thought of as being used for reading dimmer / volume knobs

They turn the signals they recieve into numbers between 0 - 1023

Accessed in code with analogRead(#);

# is the pin number (0 - 5)



# ARDUINO

This code reads Analog In Pin 0, waits 15 milliseconds, and then reads it again

A0 will recieve signals between 0 - 1023 and save those to the variable called value

But what if we need that number to correspond to a different range of values like 0 - 180 for our servo motor?

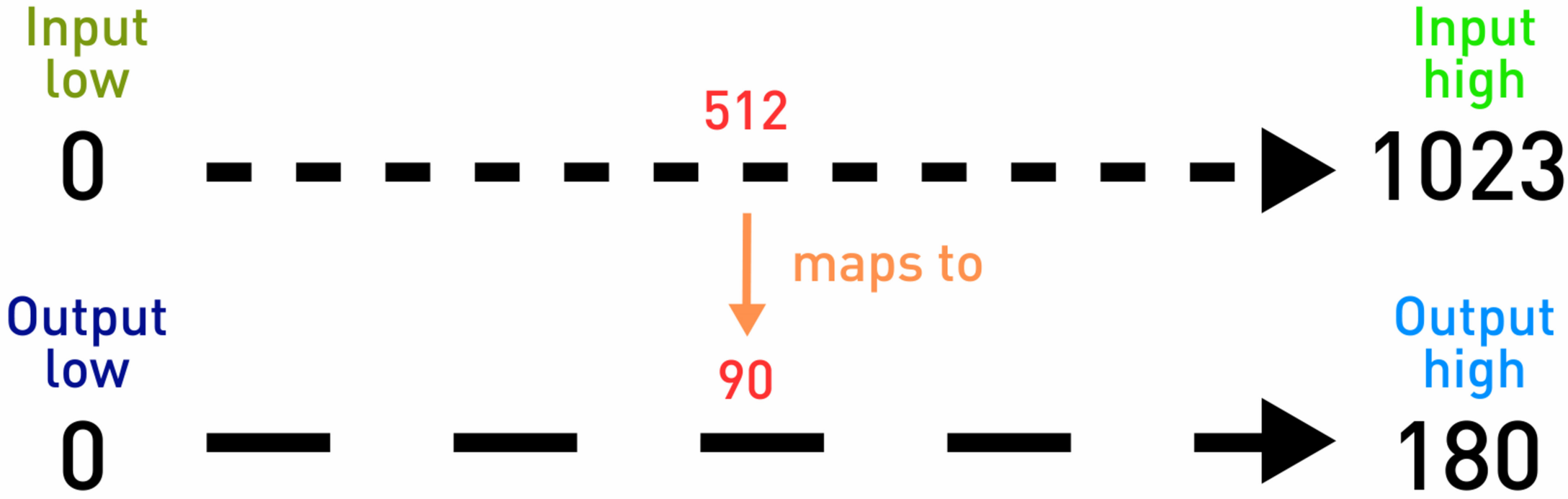
map(#, inputLow, inputHigh, outputLow, outputHigh);

sketch_mar2a.ino

```
1   void setup() {
2   }
3
4   void loop() {
5     //Read Analong In pin 0
6     int value = analogRead(A0);
7     //Map the reading to a number between 0-180
8     value = map(value, 0, 1023, 0, 180);
9     //Wait 15ms and do it again
10    delay(15);
11  }
12
```

# Analog In pins read power between GND and 5v linearly

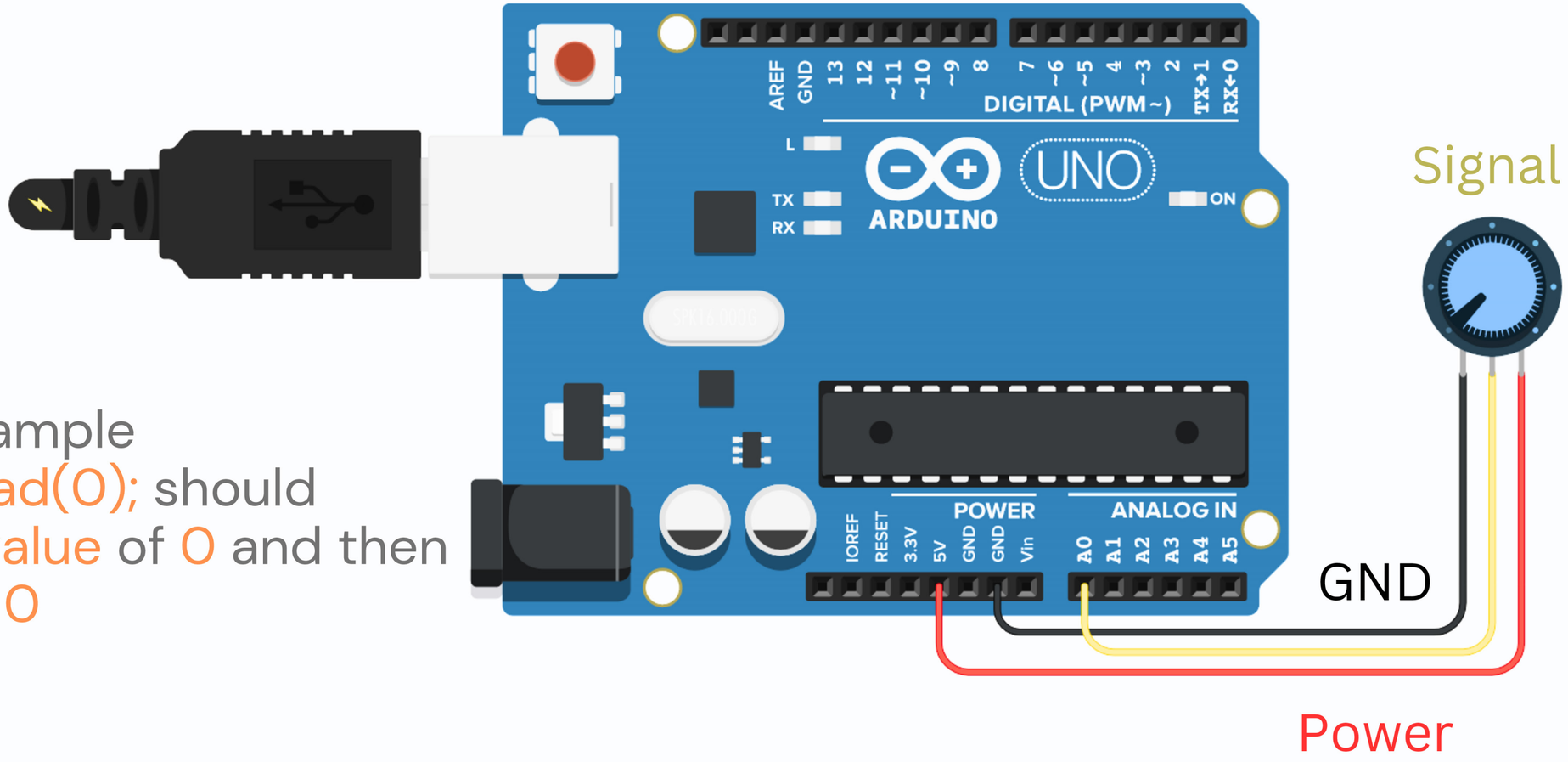
# map( ) lets you take one range of values and scale them to match another



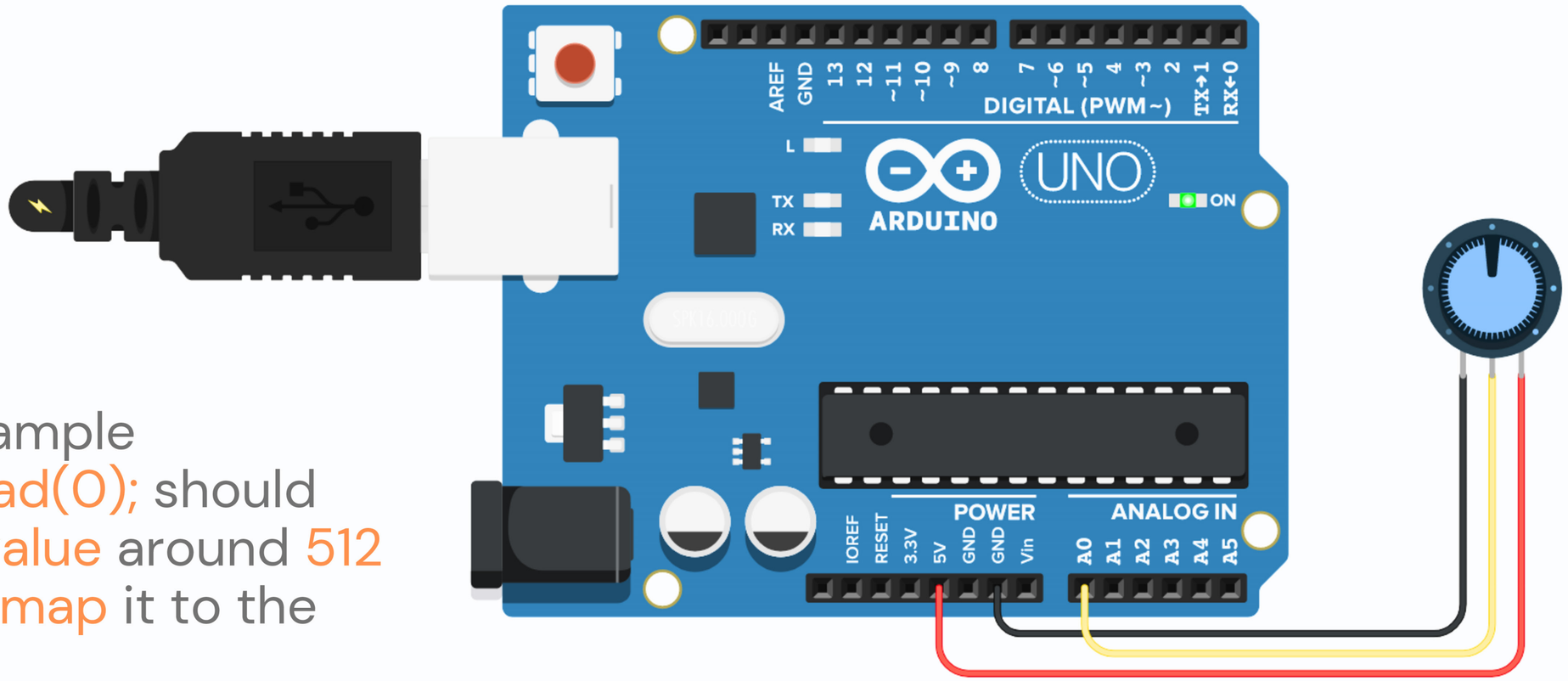

```
map(value, inputLow, inputHigh, outputLow, outputHigh);
```

# ARDUINO

In this example analogRead(0); should return a value of 0 and then map it to 0



# ARDUINO




In this example analogRead(0); should return a value around 512 and then map it to the value 90

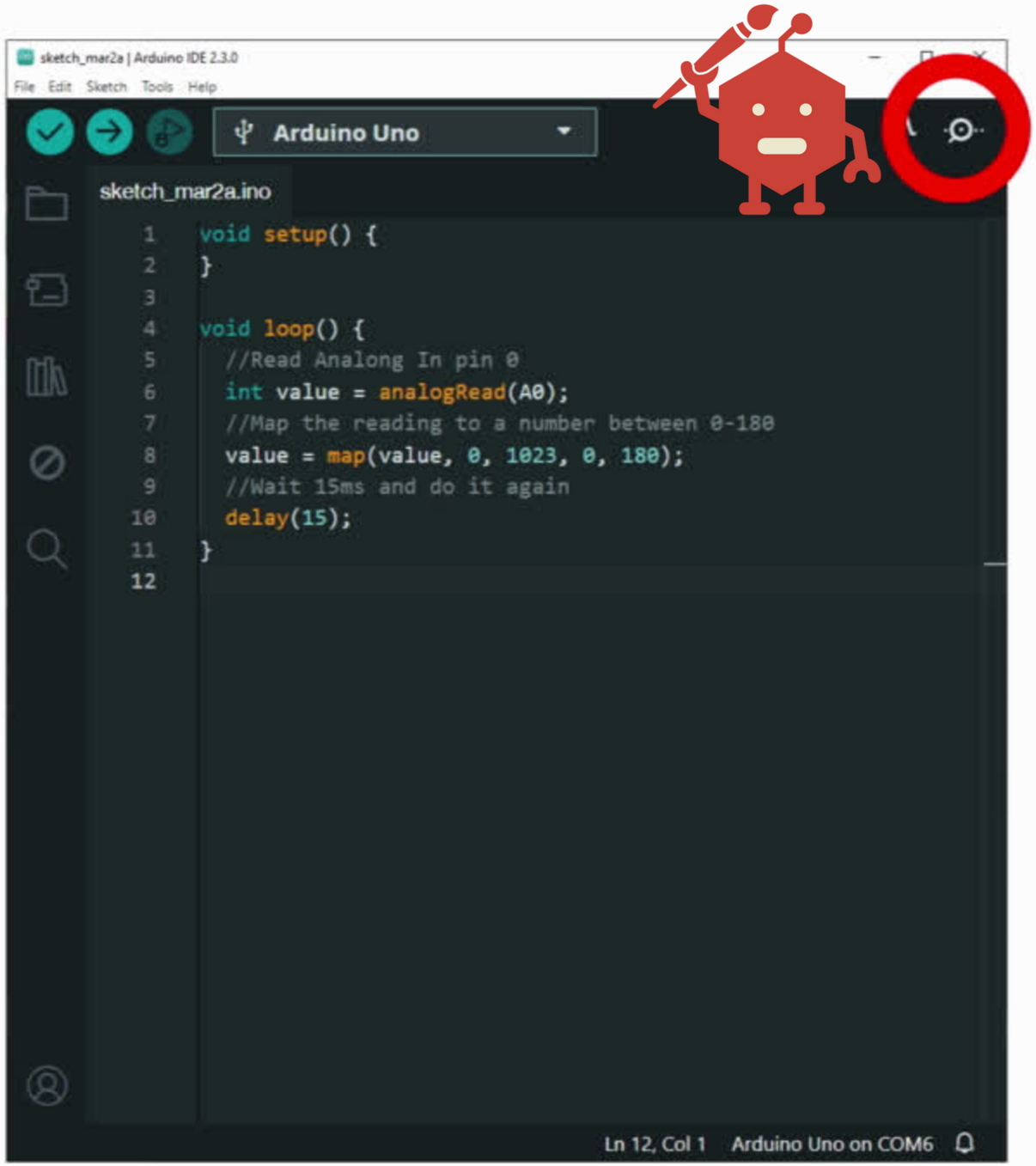
# “HOW DO I KNOW WHAT THE HELL IT’S DOING?”

# GOOD QUESTION!

The Arduino can output messages to the computer using a method called Serial [or serial bus, serial port, serial comms - like everything it can have a lot of names. Serial is easiest]

The Arduino IDE has a Serial Monitor - a window where you can display the Serial messages the Arduino is sending.

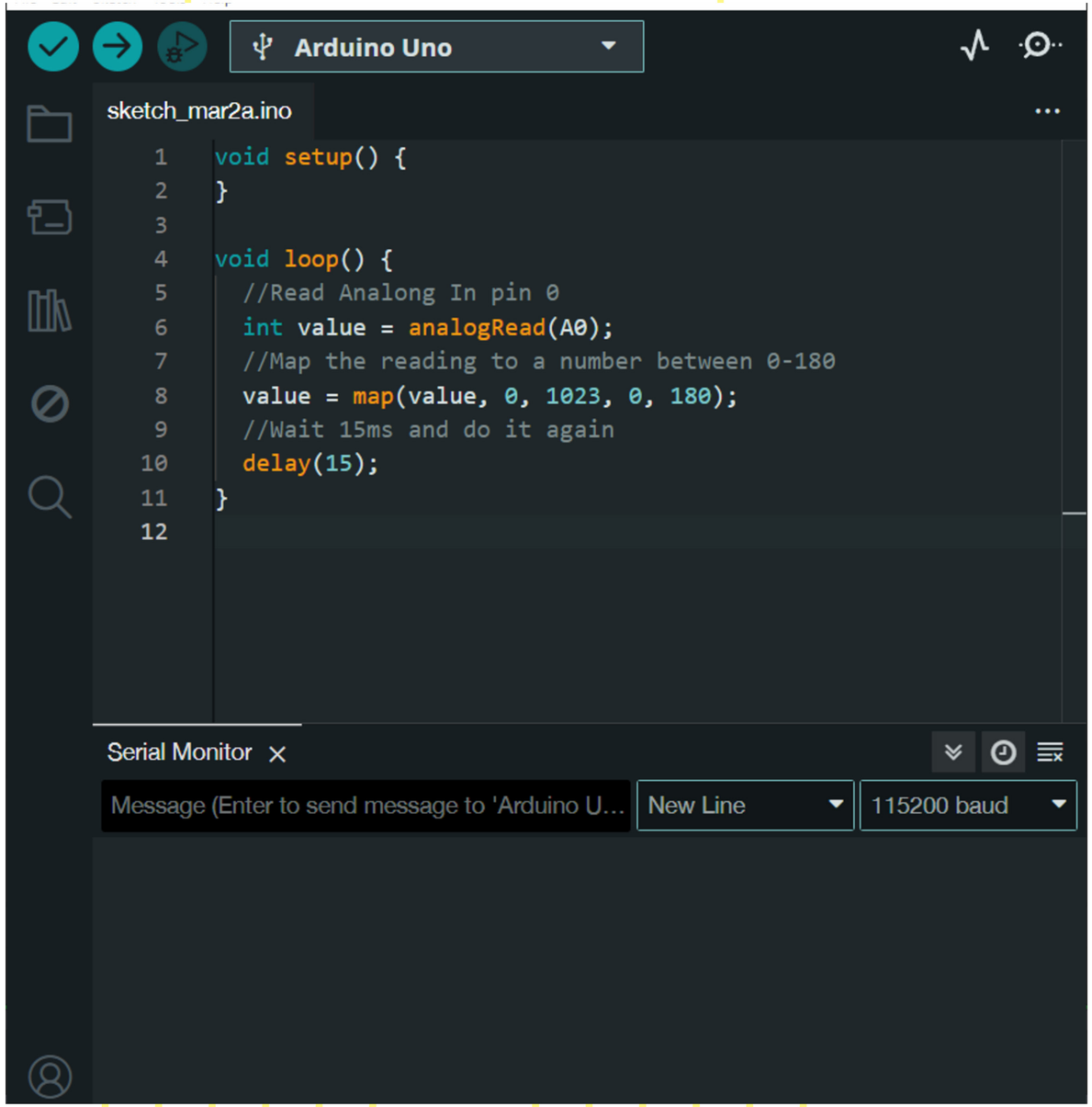
You will find it by pressing the magnifying glass icon in the top right of your IDE



# AH, THERE IT IS!

Once you press the magnifying glass you will see a window appear at the bottom of your code like the one pictured here.

It doesn’t have any data yet though.



# “HOW DO I MAKE IT SHOW THE NUMBERS???!?!?”

# CALM DOWN!

Like all our code, we have to let the Arduino know that we want to use the Serial functionality.

Serial.begin(#); tells the Arduino we want to use the serial port.

The numbers we can use here are based on old communications technology. They will make no sense to you, and you never need to understand them. You need to use one that will work, however.

I recommend sticking to 115200 it is fast and reliable.

sketch_mar2a.ino

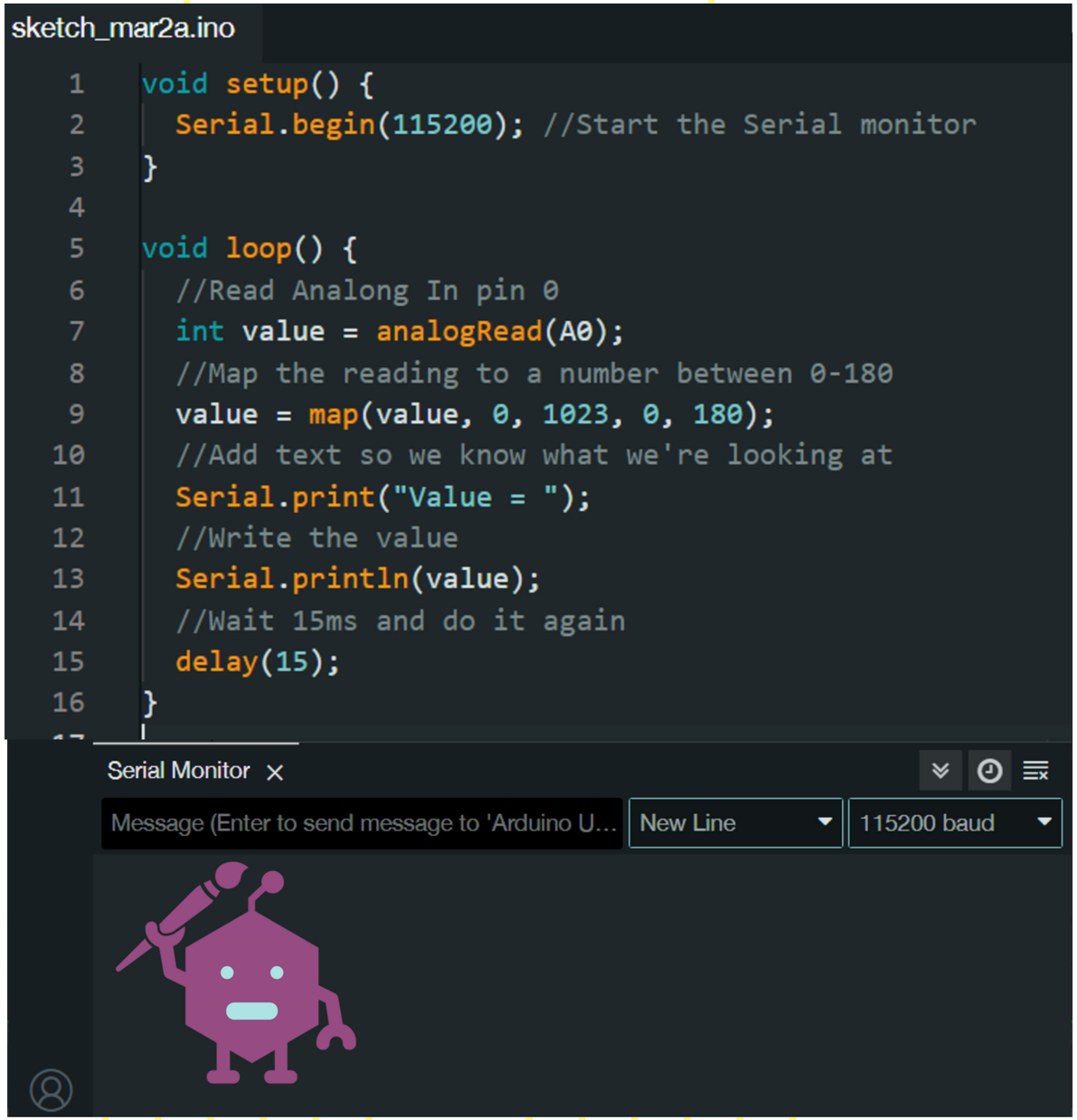
```
1   void setup() {
2     Serial.begin(115200); //Start the Serial monitor
3   }
4
5   void loop() {
6     //Read Analong In pin 0
7     int value = analogRead(A0);
8     //Map the reading to a number between 0-180
9     value = map(value, 0, 1023, 0, 180);
10    //Add text so we know what we're looking at
11    Serial.print("Value = ");
12    //Write the value
13    Serial.println(value);
14    //Wait 15ms and do it again
15    delay(15);
16  }
17
```

# NOW WHEN YOU RUN THIS CODE...

A list of values will start appearing in the Serial Monitor.

Now we know what's going on!

We can use this information to drive interaction in our system and read from all different types of sensors.



# BRINGING IT ALL TOGETHER

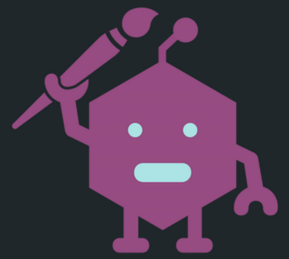
We can simply add a line at the

sketch_mar2a.ino

```
1   void setup() {
2     Serial.begin(115200); //Start the Serial monitor
3   }
4
5   void loop() {
6     //Read Analong In pin 0
7     int value = analogRead(A0);
8     //Map the reading to a number between 0-180
9     value = map(value, 0, 1023, 0, 180);
10    //Add text so we know what we're looking at
11    Serial.print("Value = ");
12    //Write the value
13    Serial.println(value);
14    //Wait 15ms and do it again
15    delay(15);
16  }
```

Serial Monitor ×

Message (Enter to send message to 'Arduino U... | New Line | 115200 baud



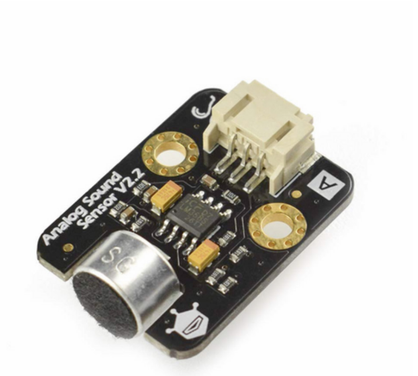
# “OKAY, BUT WHAT ABOUT SENSORS”

# SENSOR TIME!

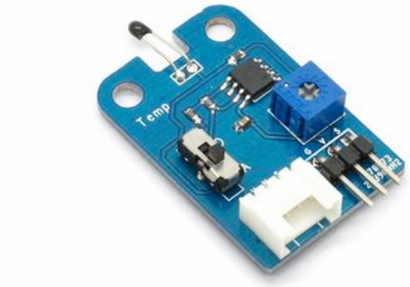
There are a whole heap of sensors we can connect to our Arduino: some analog, some digital

In almost all cases, we will use the same (or similar) code for analog sensors and the same code for digital sensors

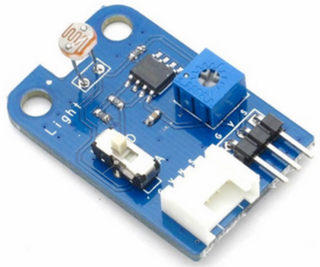
They will just have one line of difference




Audio sensor




Temperature sensor




Light sensor

# NOTE THE SIMILARITIES

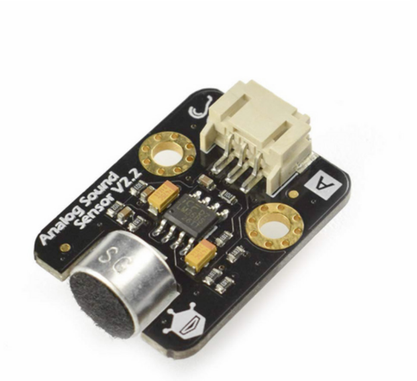
All of these sensors have a three pins we can use jumper wires to connect to

G = GROUND

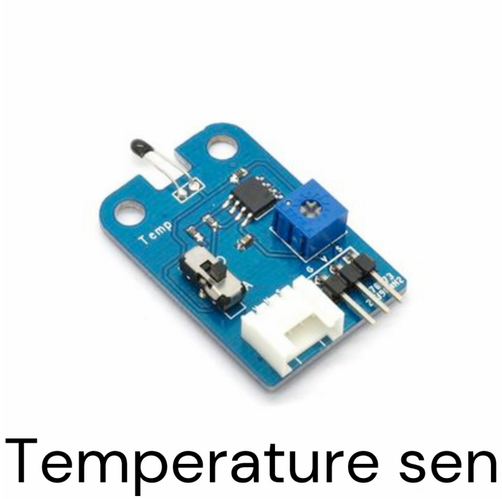
V = VOLTAGE

S = SIGNAL

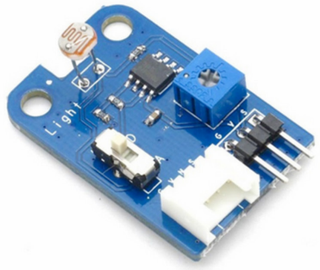
Ground and voltage are self-explanatory, and our code is reading from A0, so let's go ahead and connect the signal pin to A0




Audio sensor




Temperature sensor




Light sensor

# WORKING WITH ANALOG SENSORS

Simple 1-to-1 example:

- Servo location is set according to mapped reading

This code will map **any** analog sensor reading to the range of movement for a servo

Try it with different sensors

```
#include <Servo.h>

Servo servo;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  servo.attach(9);
}

void loop() {
  // put your main code here, to run repeatedly:
  int value = analogRead(A0);
  value = map(value, 0, 1023, 0, 180);

  Serial.print("Value = ");

  Serial.println(value);

  servo.write(value);

  delay(15);
}
```

# GETTING INTO CONDITIONALS

What if we want the Servo to perform a set movement only when an analog reading exceeds a threshold we set?

We need to adjust our code to use an if statement

IF a condition is met, run the code in the curly brackets. ELSE (ie otherwise), run the code in the second {  }

```
#include <Servo.h>

Servo servo;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);
  servo.attach(9);
}

void loop() {
  // put your main code here, to run repeatedly:
  int value = analogRead(A0);

  Serial.print("Value = ");
  Serial.println(value);

  if (value > 500) {
    servo.write(90);
  } else {
    servo.write(0);
  }

  delay(15);
}
```

# DIGITISING THINGS

What about sensors that are only ON or OFF / HIGH or LOW?

Some sensor modules have the same G, V and S markings, just switch our analog input connection across to a digital pin

Just a slight change to the code...

```
#include <Servo.h>

Servo servo;

void setup() {
  // put your setup code here, to run once:
  Serial.begin(115200);

  pinMode(2, INPUT); // we need to set the pin mode this time!

  servo.attach(9);
}

void loop() {
  // put your main code here, to run repeatedly:
  int value = digitalRead(2);

  Serial.print("Value = ");
  Serial.println(value);

  if (value == 1) {
    servo.write(90);
  } else {
    servo.write(0);
  }

  delay(15);
}
```

# NEXT WEEK

We will look at breakout boards and libraries to make use of more complex inputs, like capacitive touch