



**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

HOW DO WE RUN CREATIVE ROBOTICS CLUB?

WEEKS 2 - 5: Skill acquisition

We will learn new skills, try new ideas, grow our knowledge each week

WEEKS 7 - 10: Project support

Have the things you've learned in Weeks 2 -5 got you itching to make something? Do you have assignments that need electronics or programming support?
We are here to help.

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

BUT FIRST LETS TALK ABOUT...

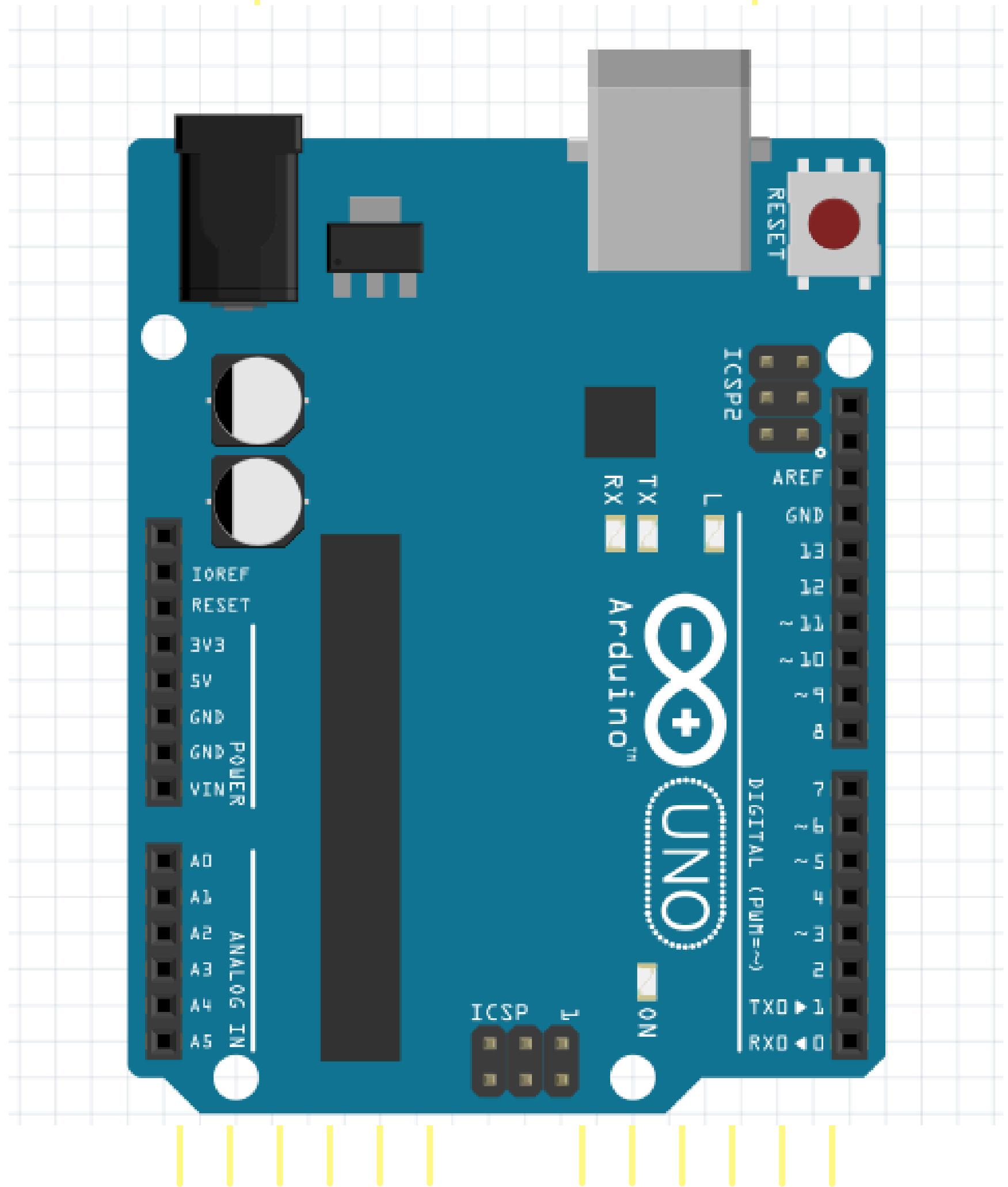
LAST WEEK

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





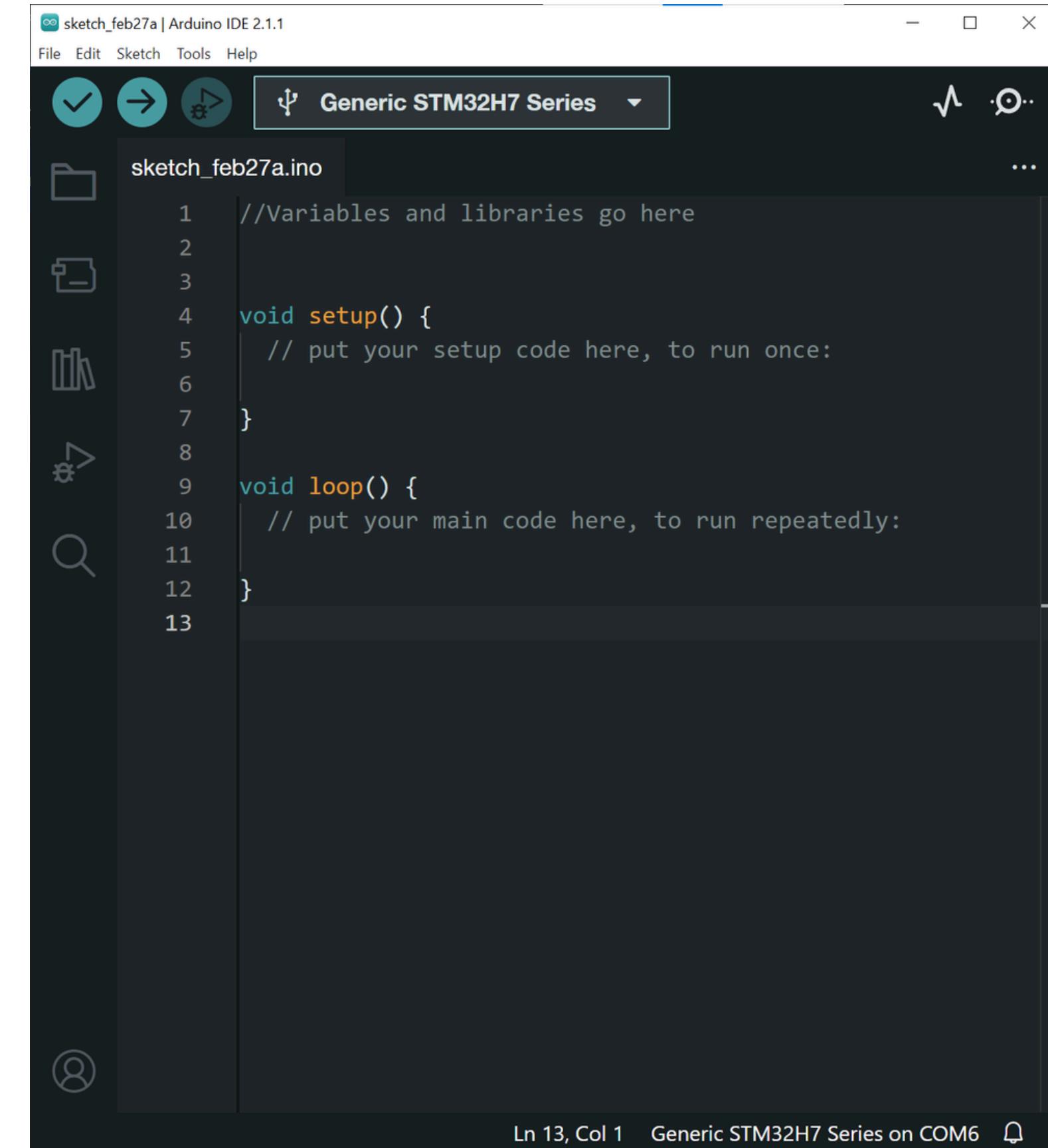
**Coding is like
cooking**

THINK OF IT LIKE A RECIPE

At the top we tell the program what ingredients we need. We call this declaring our **variables**.

In **void setup()** we tell it how to prepare those ingredients. What are the starting values for our variables?

And in **void loop()** we tell it what it is we're doing.



```
sketch_feb27a | Arduino IDE 2.1.1
File Edit Sketch Tools Help
Generic STM32H7 Series ...
sketch_feb27a.ino
1 //Variables and libraries go here
2
3
4 void setup() {
5     // put your setup code here, to run once:
6 }
7
8
9 void loop() {
10    // put your main code here, to run repeatedly:
11 }
12
13
```

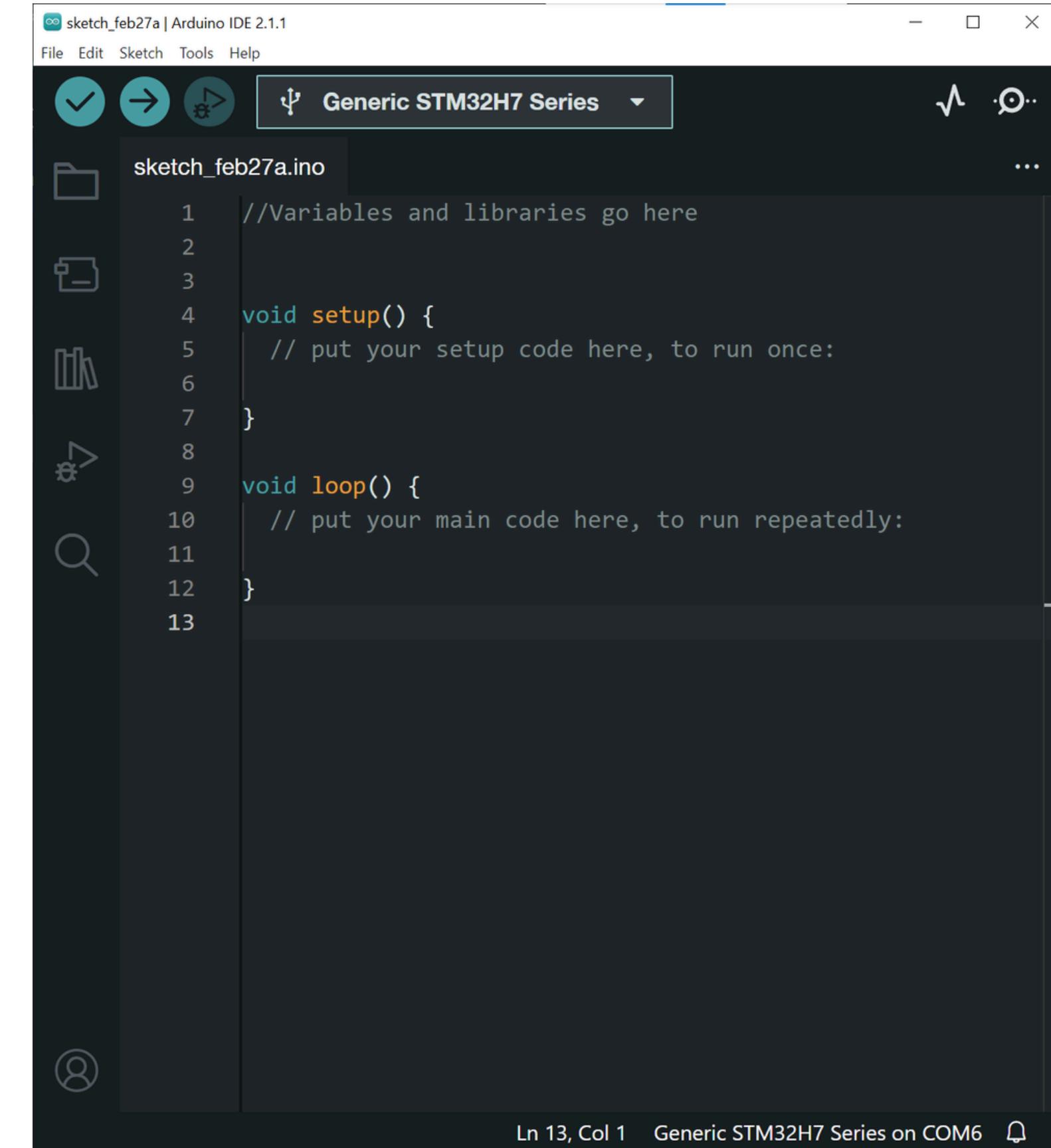
Ln 13, Col 1 Generic STM32H7 Series on COM6

THINK OF IT LIKE A RECIPE

We declare our **variables** once.

void setup() only runs at the start of our program – when the board powers on.

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



The screenshot shows the Arduino IDE interface with a dark theme. The title bar reads "sketch_feb27a | Arduino IDE 2.1.1". The toolbar includes icons for file operations (checkmark, arrow, refresh) and a dropdown menu for the board type, currently set to "Generic STM32H7 Series". The central code editor window displays the following sketch:

```
sketch_feb27a.ino
1 //Variables and libraries go here
2
3
4 void setup() {
5     // put your setup code here, to run once:
6
7 }
8
9 void loop() {
10    // put your main code here, to run repeatedly:
11
12 }
13
```

The status bar at the bottom right indicates "Ln 13, Col 1 Generic STM32H7 Series on COM6".

SERVO MOTORS

Servo will:

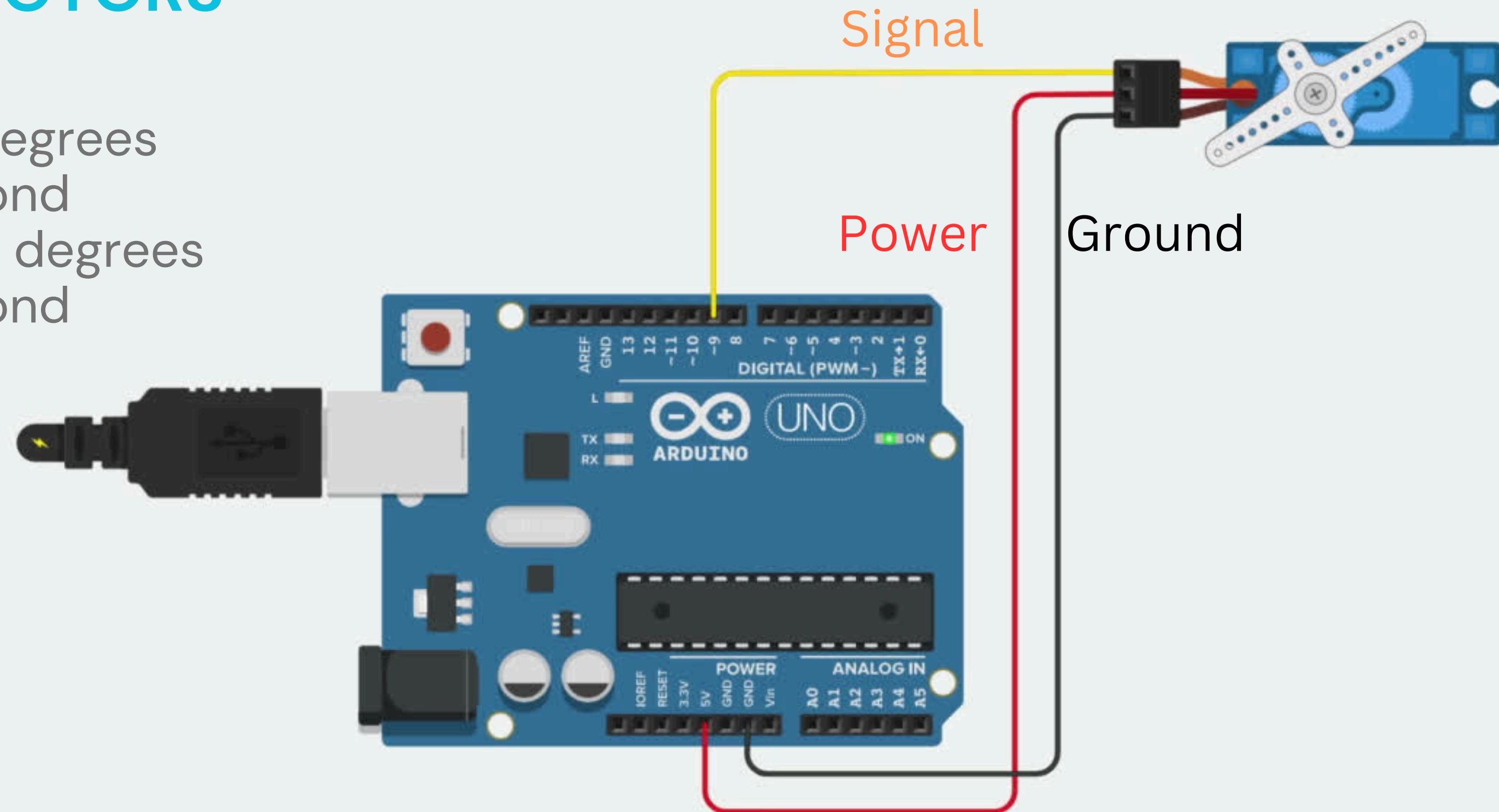
Move to 0 degrees

Pause 1 second

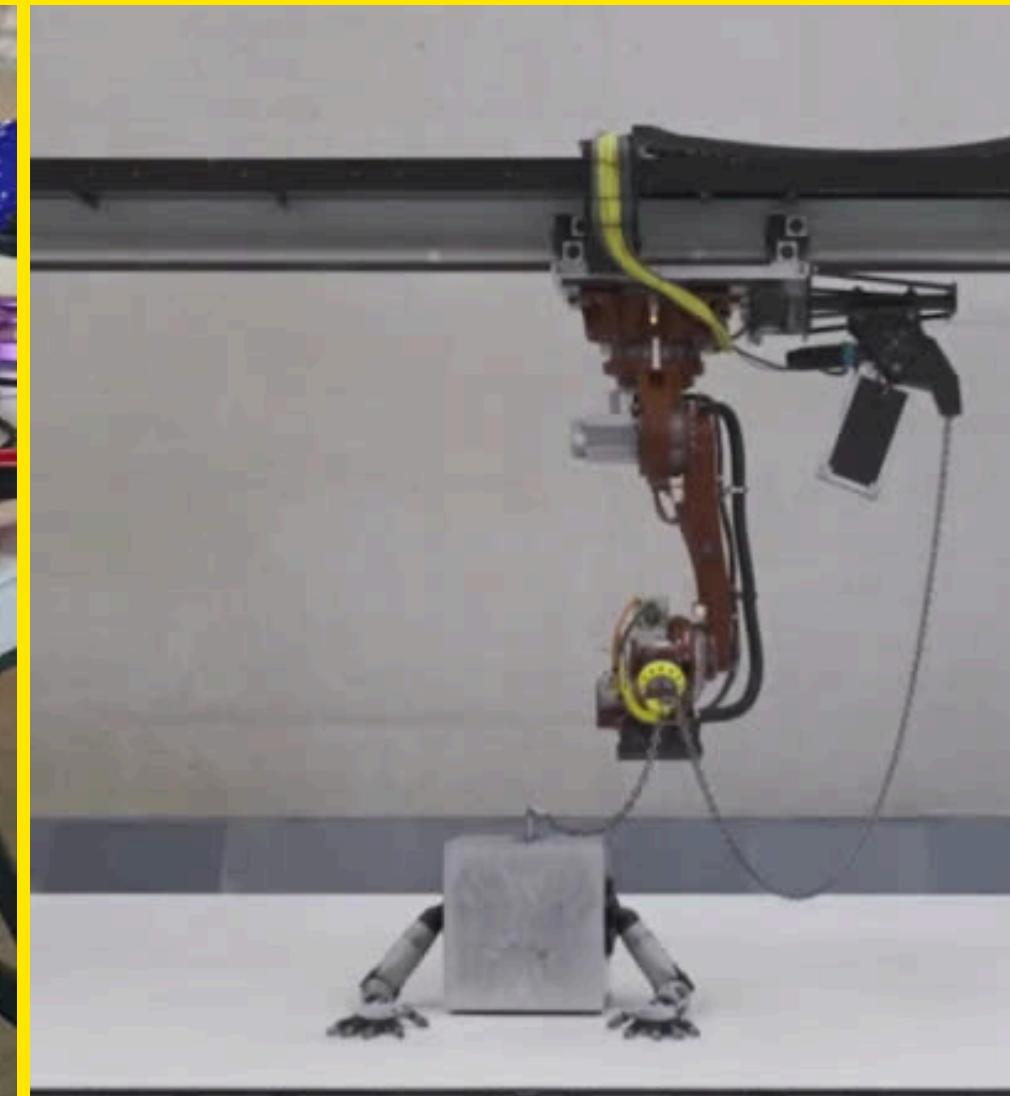
Move to 180 degrees

Pause 1 second

Repeat



WHAT ARE WE DOING TODAY AT THE CREATIVE ROBOTICS CLUB?



SENSORS + CONTROL

ARDUINO

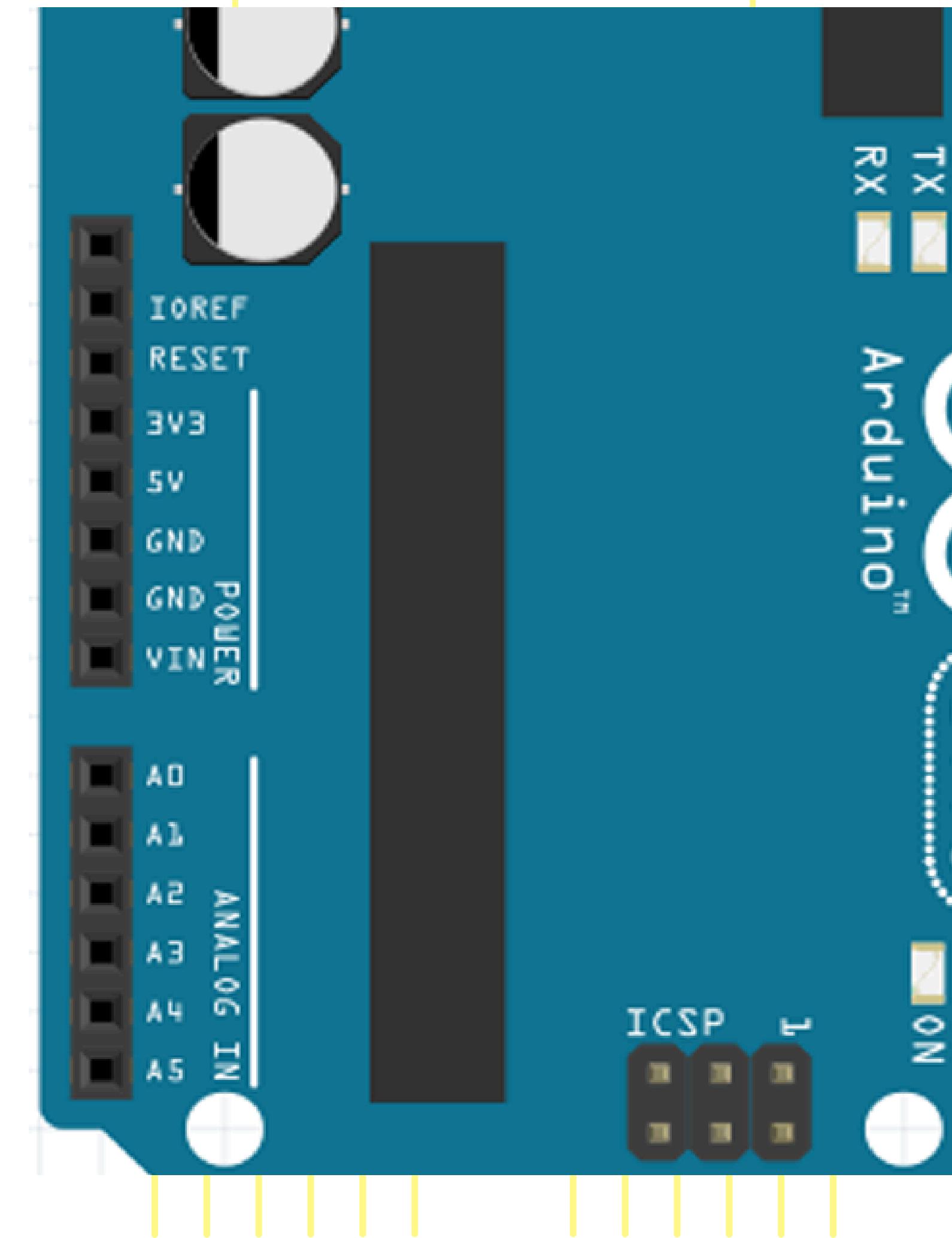
Analog In pins are located in the bottom left

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

They turn the signals they receive 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

AO will receive 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);**

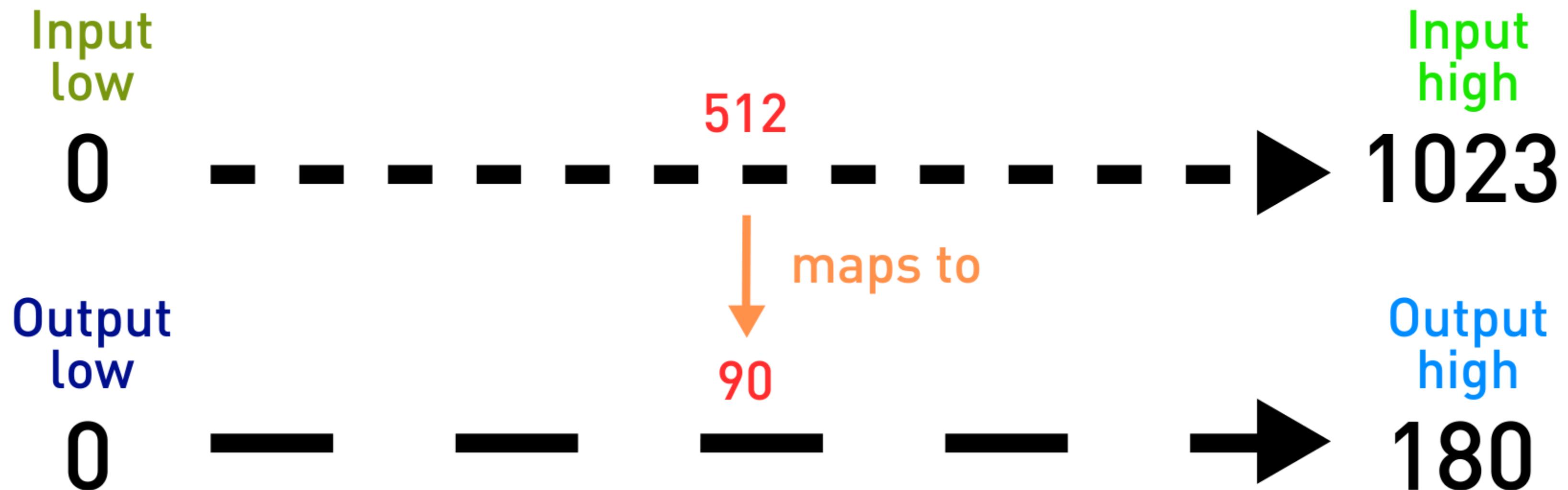
sketch_mar2a.ino

```
1 void setup() {  
2 }  
3  
4 void loop() {  
5     //Read Analog 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



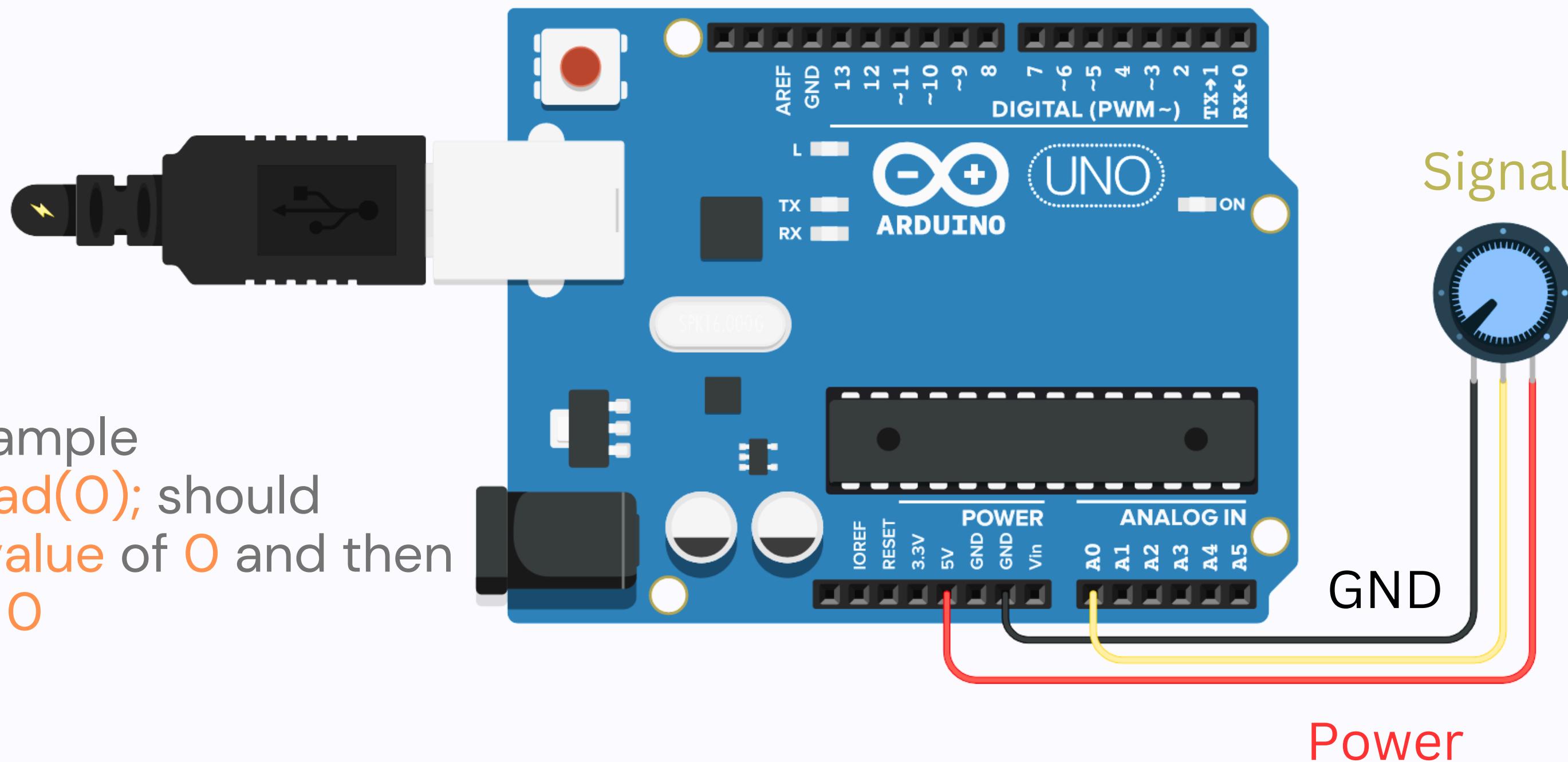
map() lets you take one range of values and scale them to match another



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

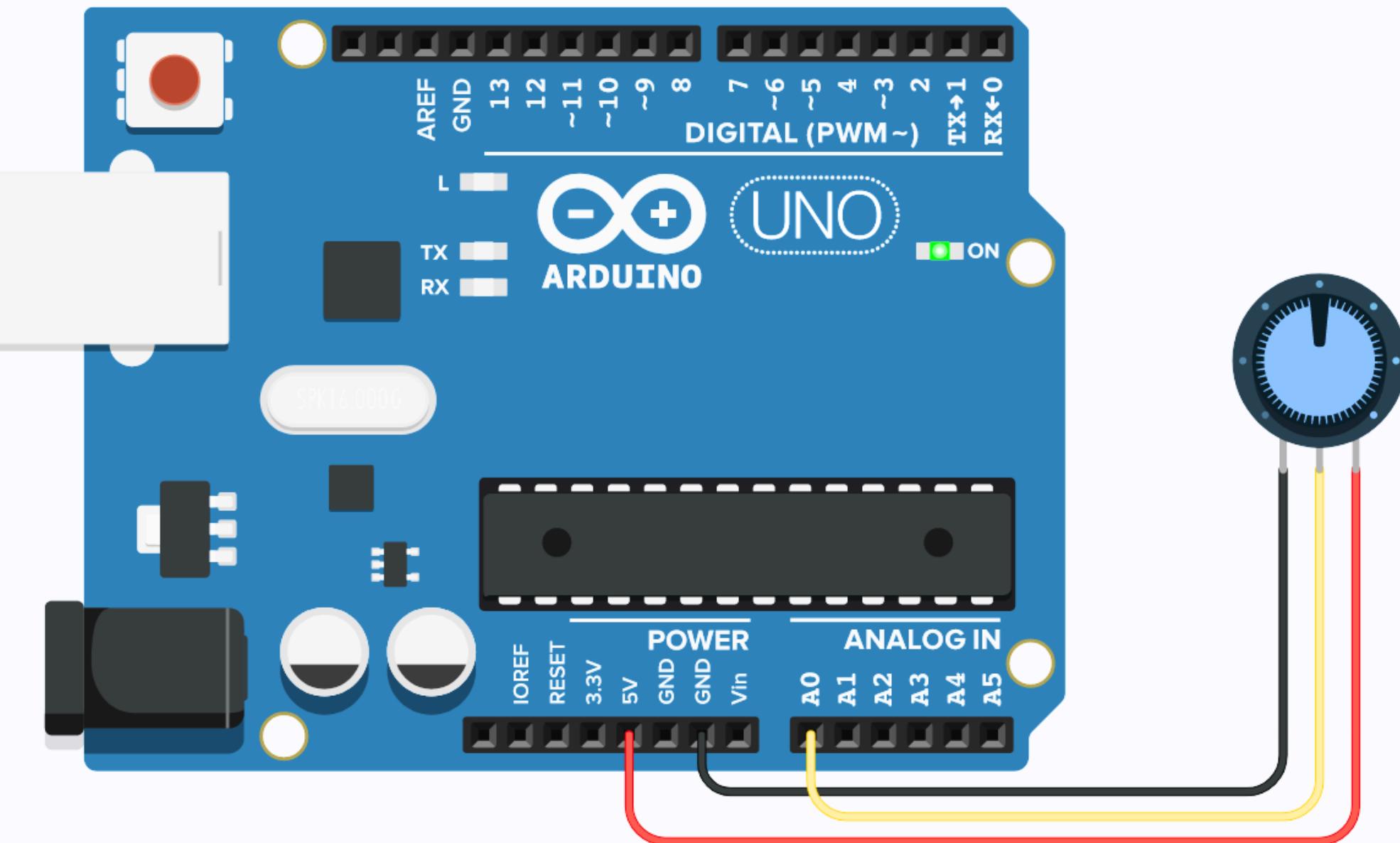
ARDUINO

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?”

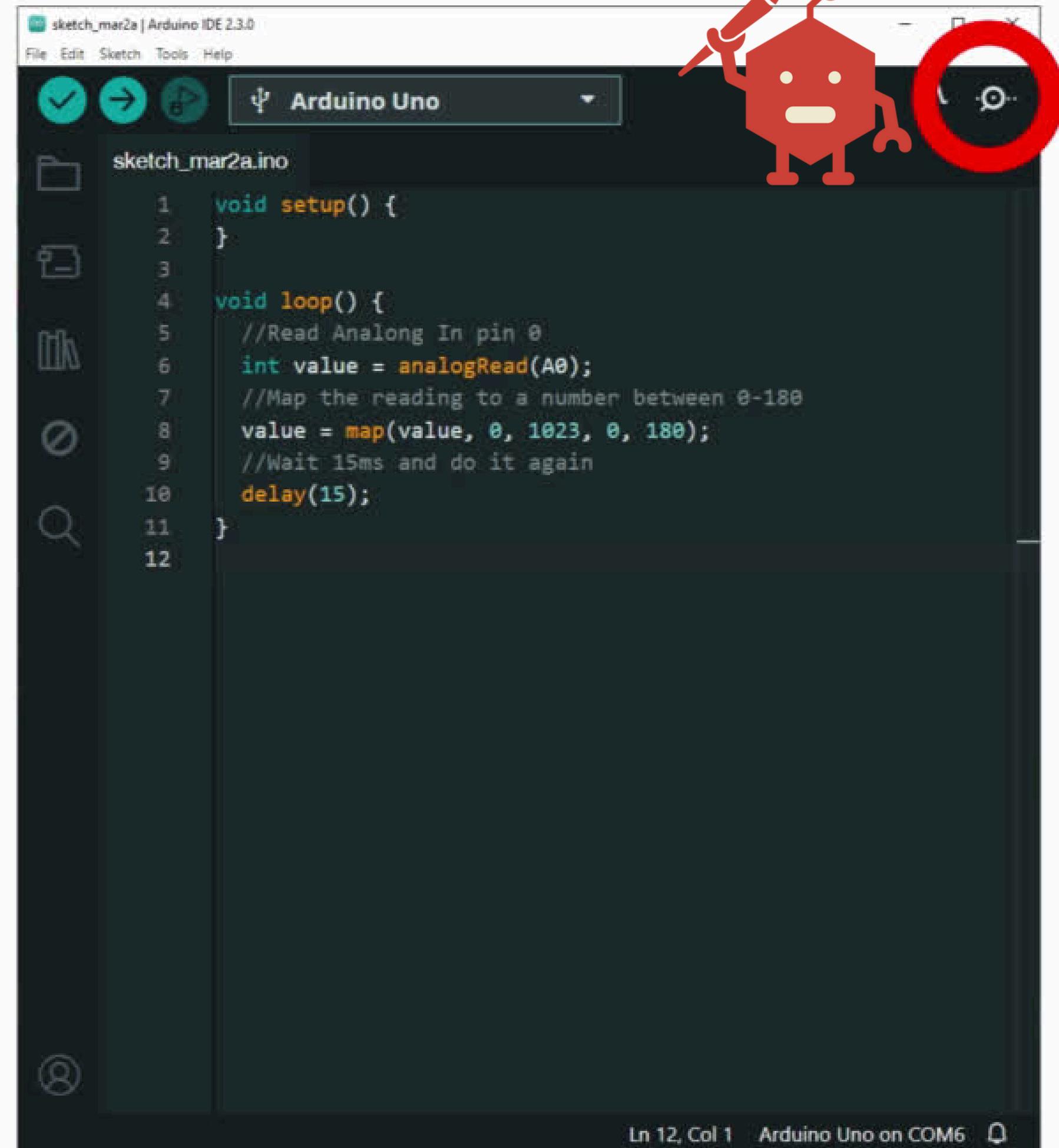


GOOD QUESTION!

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



The screenshot shows the Arduino IDE interface. The title bar reads "sketch_mar2a | Arduino IDE 2.3.0". The toolbar includes icons for file operations, a checkmark, a green arrow, a blue circular arrow, and a dropdown menu set to "Arduino Uno". The left sidebar contains icons for file, folder, and search. The main code editor window displays the following sketch:

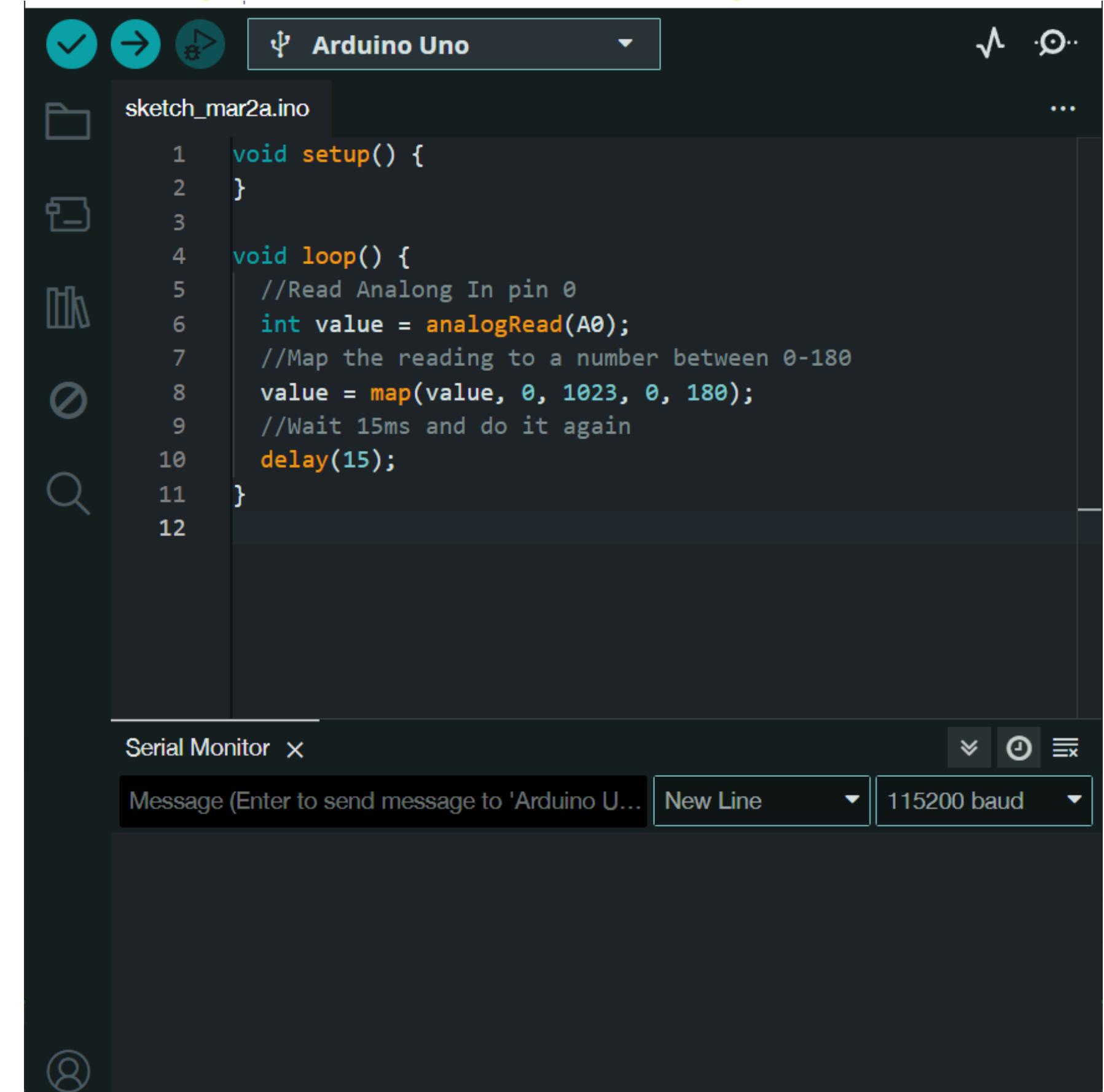
```
1 void setup() {
2 }
3
4 void loop() {
5     //Read Analog In pin 0
6     int value = analogRead(A0);
7     //Map the reading to a number between 0-180
8     value = map(value, 0, 1023, 0, 180);
9     //Wait 15ms and do it again
10    delay(15);
11 }
12 }
```

At the bottom right of the code editor, the status bar shows "Ln 12, Col 1" and "Arduino Uno on COM6". A red cartoon robot character is positioned in the top right corner of the IDE window, holding a paintbrush. A large red circle highlights the magnifying glass icon in the top right corner of the IDE frame.

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.

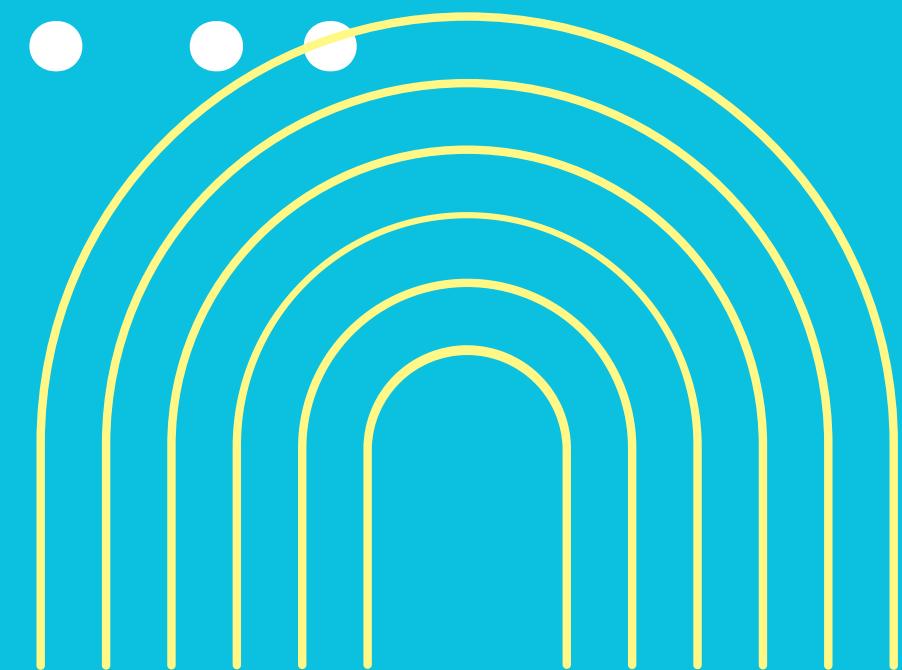


The screenshot shows the Arduino IDE interface. At the top, there are three circular icons: a checkmark, a right-pointing arrow, and a gear. Next to them is a USB port icon followed by the text "Arduino Uno". To the right of the Uno icon are three small yellow vertical bars. On the far right, there are icons for a magnifying glass, a refresh, and three dots. Below these icons, a file tree on the left shows a folder named "sketch_mar2a.ino". The main code editor area contains the following code:

```
1 void setup() {  
2 }  
3  
4 void loop() {  
5 //Read Analog 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
```

Below the code editor is a horizontal bar with the text "Serial Monitor X" and three small icons. The main window below the bar is titled "Serial Monitor" and contains a text input field with the placeholder "Message (Enter to send message to 'Arduino U...')". To the right of the input field are dropdown menus for "New Line" and "115200 baud".

“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

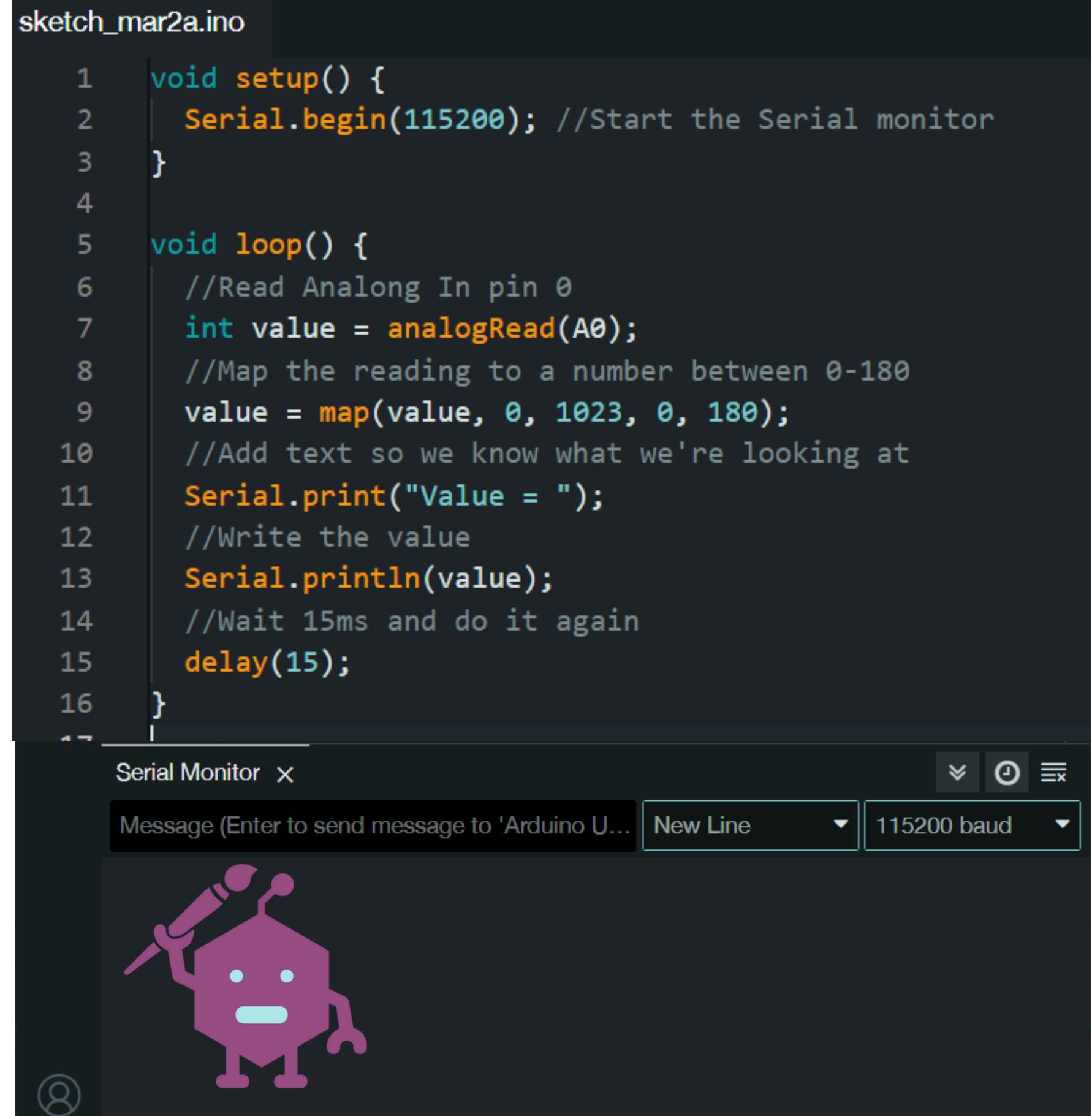
```
1 void setup() {  
2     Serial.begin(115200); //Start the Serial monitor  
3 }  
4  
5 void loop() {  
6     //Read Analog In pin 0  
7     int value = analogRead(A0);  
8     //Map the reading to a number between 0-180  
9     value = map(value, 0, 1023, 0, 180);  
10    //Add text so we know what 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.



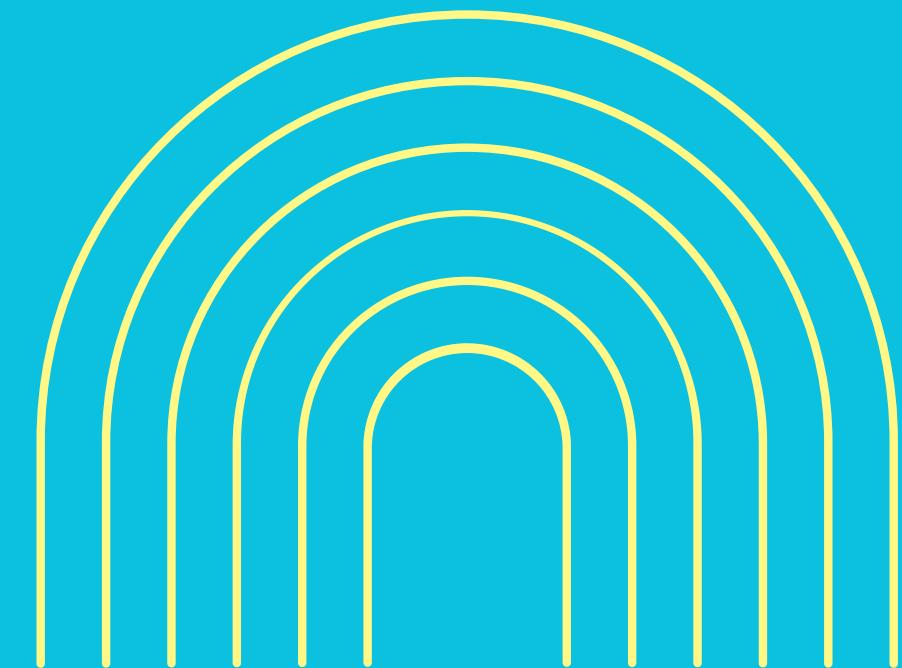
The image shows the Arduino IDE interface. The top window displays the code for 'sketch_mar2a.ino' with line numbers 1 through 16. The code reads an analog input from pin A0, maps it to a range of 0-180, adds text 'Value = ', and prints the result to the Serial Monitor. It then waits 15ms before repeating the loop. The bottom window is the 'Serial Monitor' showing a message input field and a baud rate selector set to 115200. A small purple robot icon is visible in the bottom left corner of the monitor window.

```
sketch_mar2a.ino
1 void setup() {
2     Serial.begin(115200); //Start the Serial monitor
3 }
4
5 void loop() {
6     //Read Analog 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”

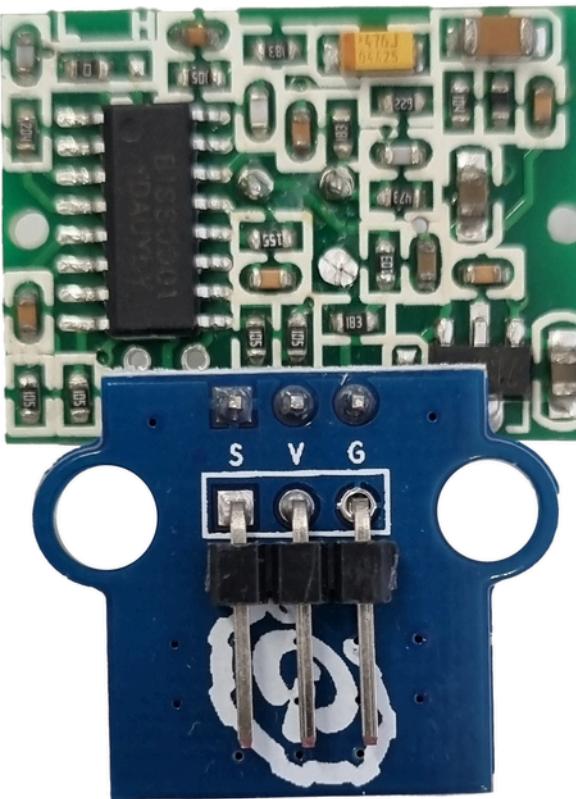
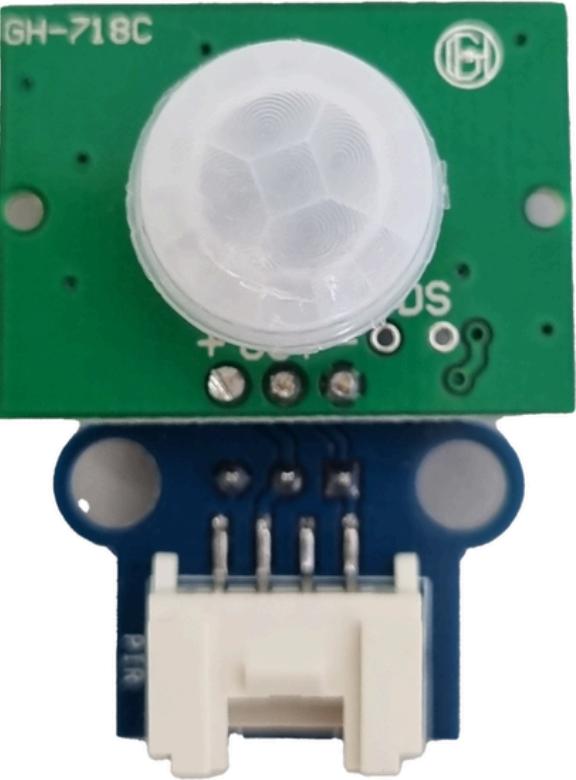


SENSOR TIME!

Not all sensors use the **Analog In** pins. The **PIR sensor** (passive infrared) detects motion and sends a digital signal.

Take a look at the markings on the sensor. You can attach wires to **S**, **V**, and **G**. Looking back over the things we've done over the last few weeks can you guess what **S**, **V** and **G** stand for?

If not how do you think you might find out?

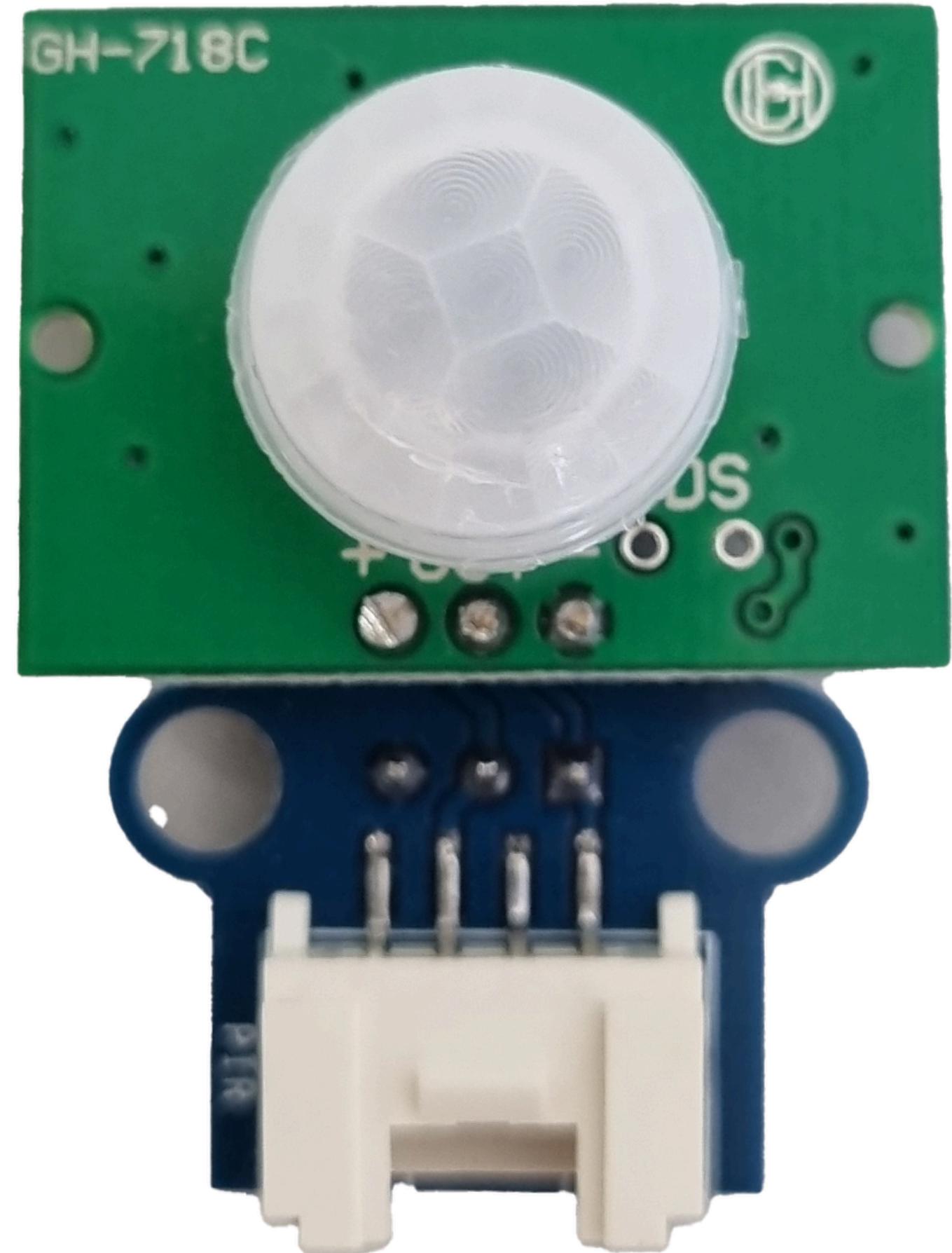


IT'S GOOGLE TIME!

The first thing we can do is use Google. Searching for **PIR sensor** and (or even just) the part number we can see on the device **GH-718C** will give us a lot of information.

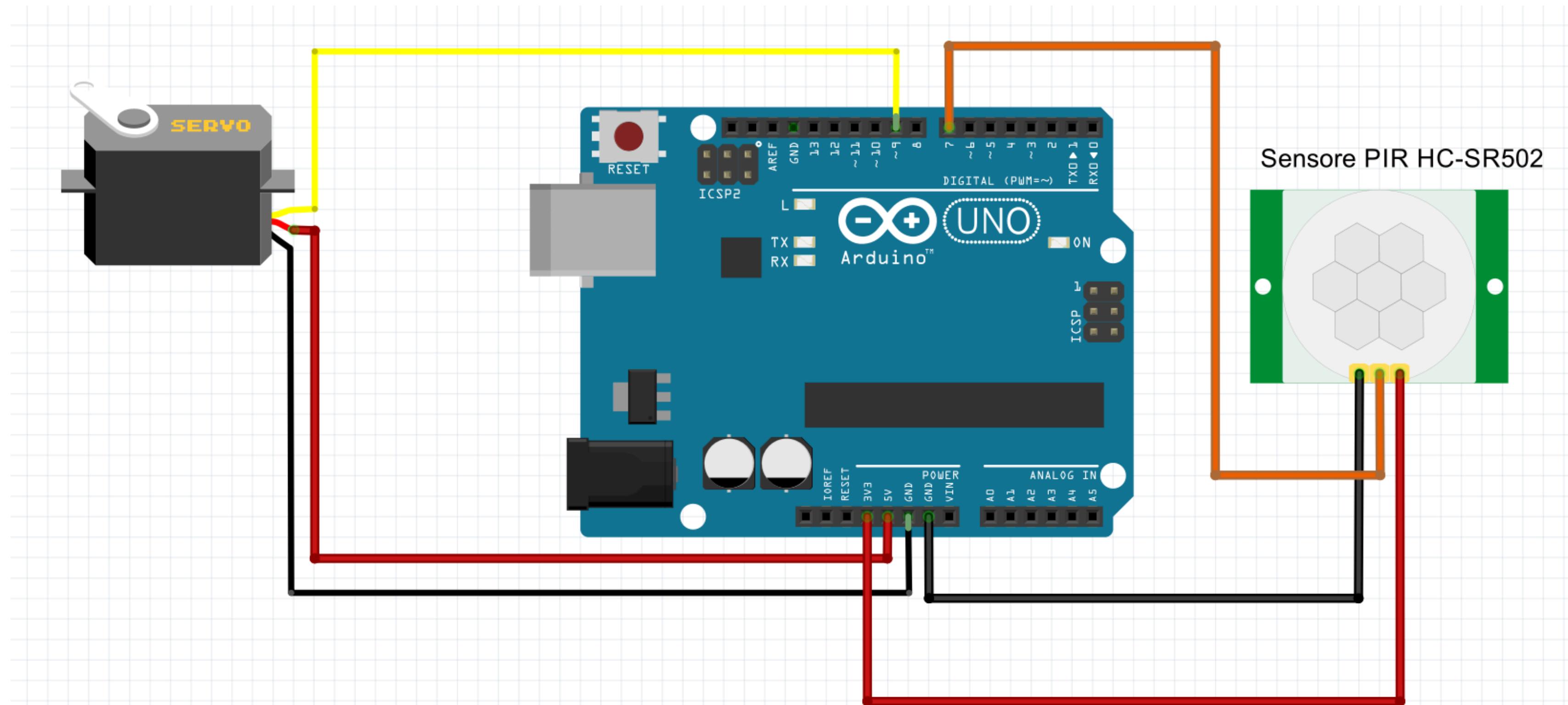
Almost every part you will use will have some information about it on the internet. It is always worth checking to see if someone else has run into the same problems, and solved them, before you.

Chat GPT is also useful, but not perfect, or this.



S = SIGNAL V = VOLTAGE and G = GND

So, now we know how to wire this sensor up!



LETS GET IT WORKING!

There's some new code in here, lets check it out.

A new variable: **bool**

Bools are either

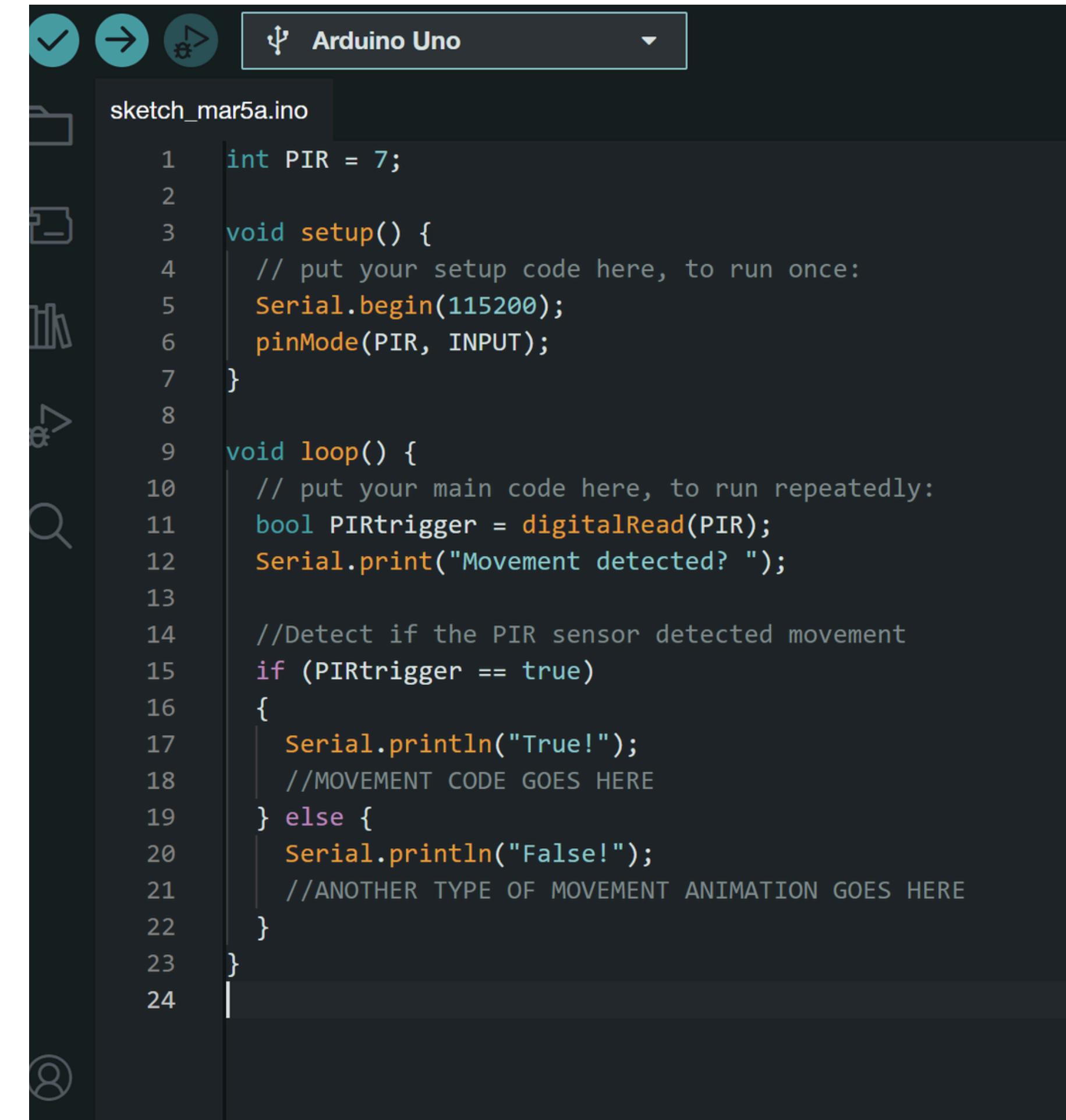
TRUE or **FALSE** / **1** or **0**

digitalRead(#);

Checks the digital pin # to see what voltage is present there.

0 = GND

1 = > **3.3v**



```
int PIR = 7;

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

void loop() {
  // put your main code here, to run repeatedly:
  bool PIRtrigger = digitalRead(PIR);
  Serial.print("Movement detected? ");

  //Detect if the PIR sensor detected movement
  if (PIRtrigger == true)
  {
    Serial.println("True!");
    //MOVEMENT CODE GOES HERE
  } else {
    Serial.println("False!");
    //ANOTHER TYPE OF MOVEMENT ANIMATION GOES HERE
  }
}
```

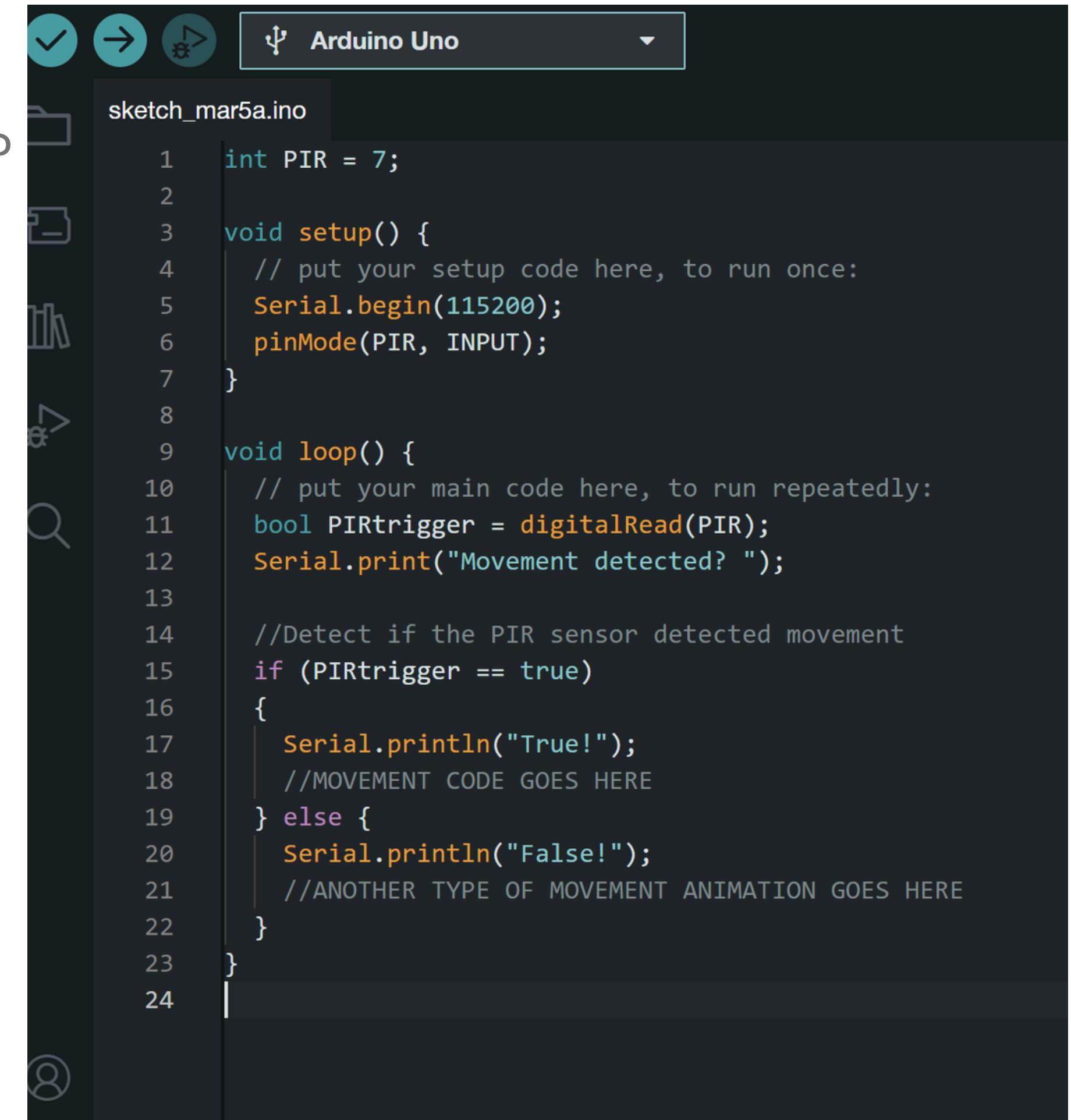
LETS GET IT WORKING!

How do we make decisions in code?
The **IF** statement. It works with
maths, but really simple maths

```
if (maths statement is true)
{
    do this stuff
}
```

Here we are checking if the
PIR sensor is working.
Is it reading 1 or 0, **TRUE** or **FALSE**?

One = sets the value of a variable.
Two == checks that it matches
right hand side,



The screenshot shows the Arduino IDE interface with the following details:

- Top Bar:** Contains icons for file operations (checkmark, arrow, refresh), a USB port icon, and the text "Arduino Uno".
- Sketch Name:** "sketch_mar5a.ino" is selected in the left sidebar.
- Code Area:** Displays the following C-like pseudocode:

```
1 int PIR = 7;
2
3 void setup() {
4     // put your setup code here, to run once:
5     Serial.begin(115200);
6     pinMode(PIR, INPUT);
7 }
8
9 void loop() {
10    // put your main code here, to run repeatedly:
11    bool PIRtrigger = digitalRead(PIR);
12    Serial.print("Movement detected? ");
13
14    //Detect if the PIR sensor detected movement
15    if (PIRtrigger == true)
16    {
17        Serial.println("True!");
18        //MOVEMENT CODE GOES HERE
19    } else {
20        Serial.println("False!");
21        //ANOTHER TYPE OF MOVEMENT ANIMATION GOES HERE
22    }
23}
24|
```