

Happy Arduino time!

with Oxram and Wendy

What is Arduino?

Arduino Board:

send instruction to the microcontroller

read inputs - light on a sensor, a finger on a button, or a Twitter message - and turn it into an output - activating a motor, turning on an LED, publishing something online

Arduino Software:

to do so you write the code in Arduino programming language on the Arduino IDE

Things you can achieve with Arduino!



Punch Activated Arm Flamethrowers (Real...

Project showcase by **Allen Pan**

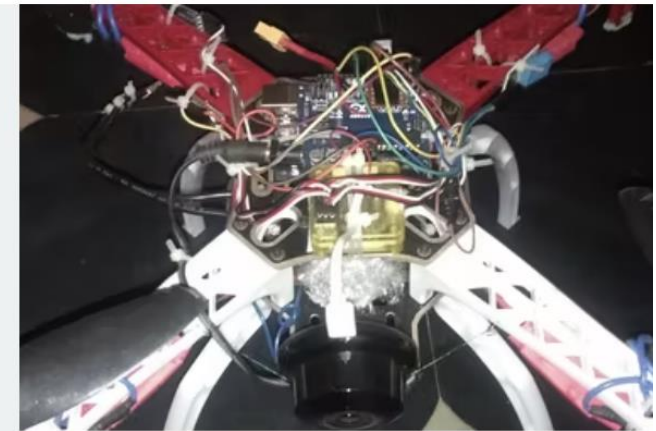
323,554 VIEWS **183** COMMENTS
1,841 RESPECTS



Portable Arduino Temp/Humidity Sensor

Project tutorial by **ThothLoki**

329,460 VIEWS **174** COMMENTS
459 RESPECTS



Autopilot Drone

Project in progress by **suhaskd**

85,709 VIEWS **233** COMMENTS
210 RESPECTS

The quickest way to learn Arduino?

Play with it!

- 1) Read a potentiometer/the voltage across a resistance and print the results in the Arduino Serial Monitor
- 2) Make an LED blink and fade
- 3) Make a servo move!
- 4) Make a stepper motor move!
- 5) Make 6 stepper motors move! (with Arduino mega and RAMPS1.4)

Download

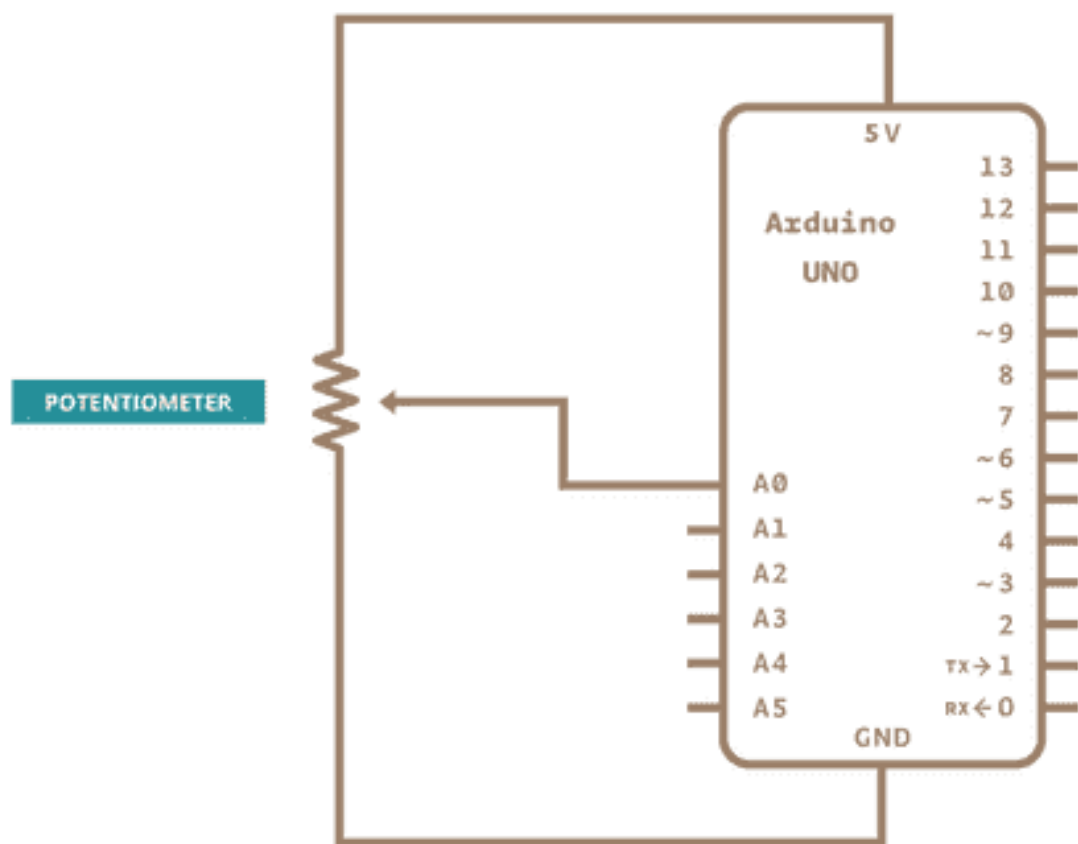
Download Arduino IDE from
<https://www.arduino.cc/en/software>

1 Analog Read Serial

code
function
variable
structure

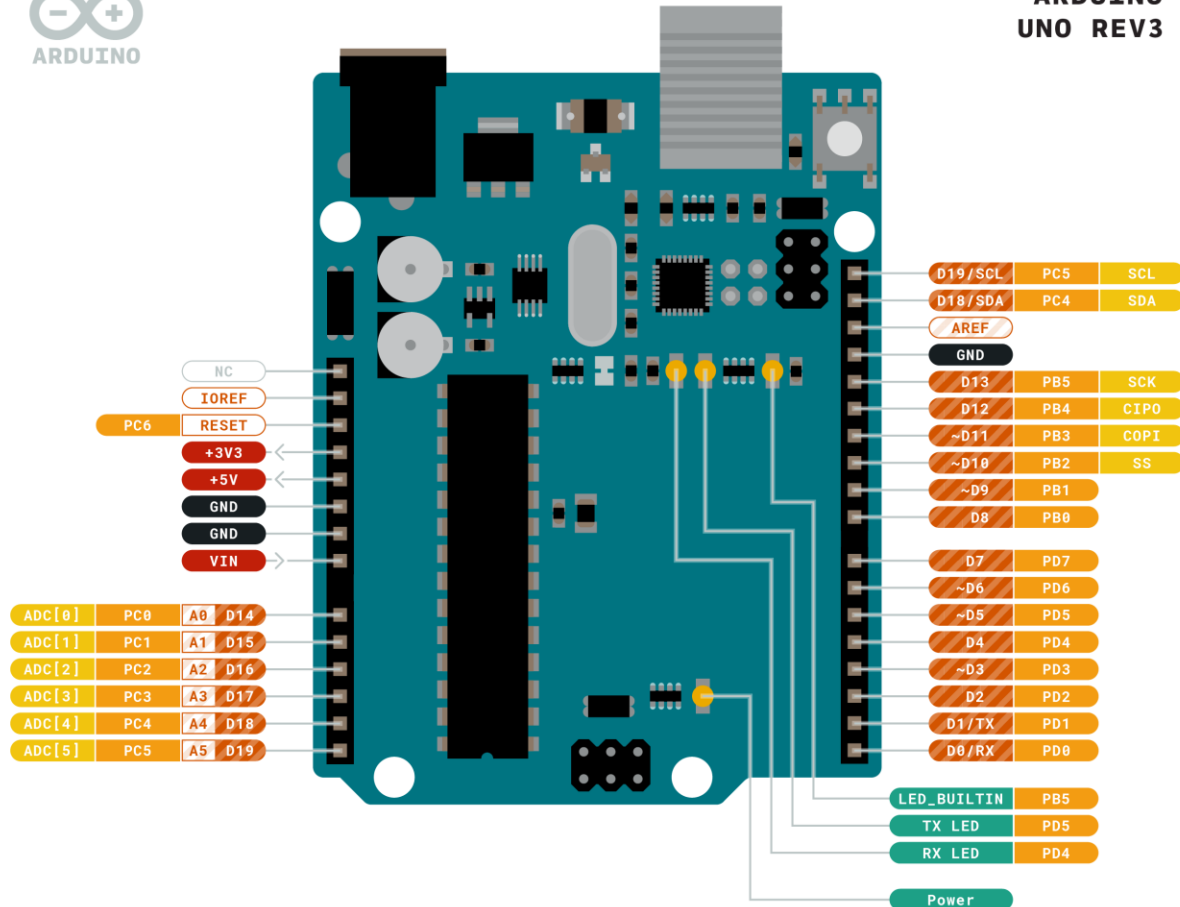
```
// the setup routine runs once when you press reset:
void setup() {
  // initialize serial communication at 9600 bits per second:
  Serial.begin(9600);
}

// the loop routine runs over and over again forever:
void loop() {
  // read the input on analog pin 0:
  int sensorValue = analogRead(A0);
  // print out the value you read:
  Serial.println(sensorValue);
  delay(1);          // delay in between reads for stability
}
```

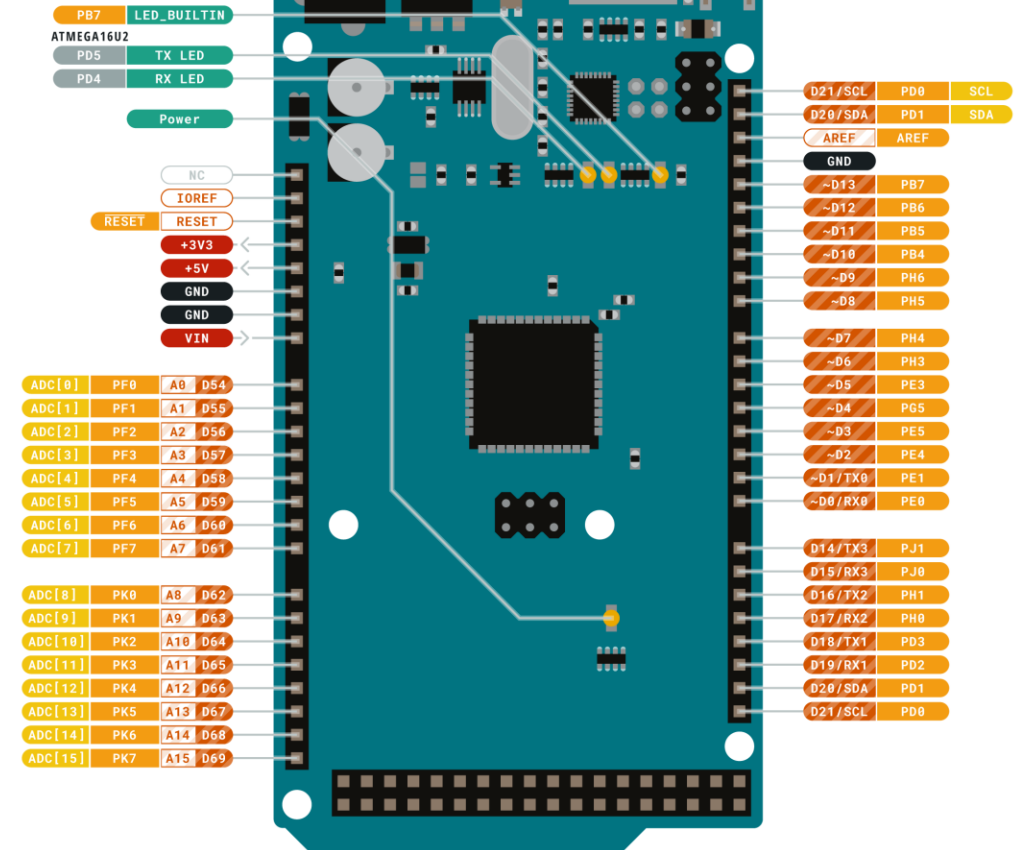




ARDUINO UNO REV3



ARDUINO MEGA 2560 REV3



- Ground
- Internal Pin
- Digital Pin
- Microcontroller's Port
- Power
- SWD Pin
- Analog Pin
- LED
- Other Pin
- Default

ARDUINO.CC



This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/4.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

- Ground
- Internal Pin
- Digital Pin
- Microcontroller's Port
- Power
- SWD Pin
- Analog Pin
- LED
- Other Pin
- Default

ARDUINO.CC



This work is licensed under the Creative Commons Attribution-ShareAlike 4.0 International License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-sa/4.0/> or send a letter to Creative Commons, PO Box 1866, Mountain View, CA 94042, USA.

Now can you convert the results in voltages?

- Convert the analog reading (which goes from 0 - 1023) to a voltage (0 - 5V)
- `float voltage = sensorValue * (5.0 / 1023.0);`
- `Serial.println(voltage);`

2 Blinking and fading LED

```
// 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
}
```

Not every pin can be used!
only PWM(Pulse Width Modulation) pins can be
used to do analogwrite

- Uno, Nano, Mini
 - 3, 5, 6, 9, 10, 11
 - 490 Hz (pins 5 and 6: 980 Hz)
-
- Mega
 - 2 - 13, 44 - 46
 - 490 Hz (pins 4 and 13: 980 Hz)

```
int led = 9;           // the PWM pin the LED is attached to
int brightness = 0;    // how bright the LED is
int fadeAmount = 5;    // how many points to fade the LED by

// the setup routine runs once when you press reset:
void setup() {
  // declare pin 9 to be an output:
  pinMode(led, OUTPUT);
}

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

  // change the brightness for next time through the loop:
  brightness = brightness + fadeAmount;

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

3 servo

- we use a servo in the gripper
- now we have to include a library called “Servo”. To check if a library(Servo is in-built), we go to Sketch->Include Library->Manage libraries
- within this library, you can have the functions:
- `attach()`; `write()`; `writeMicroseconds()`; `read()`; `attached()`; `detach()`

You can also find the code in file-examples-servo-sweep

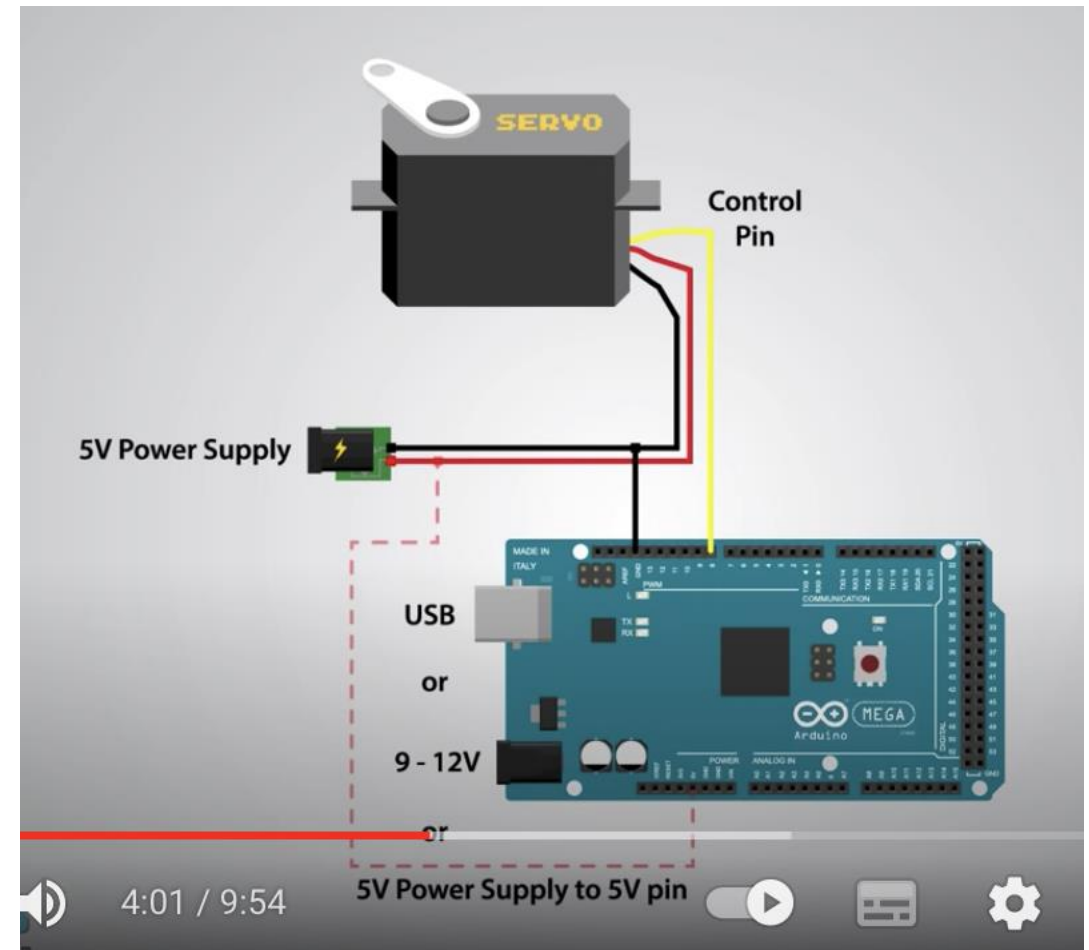
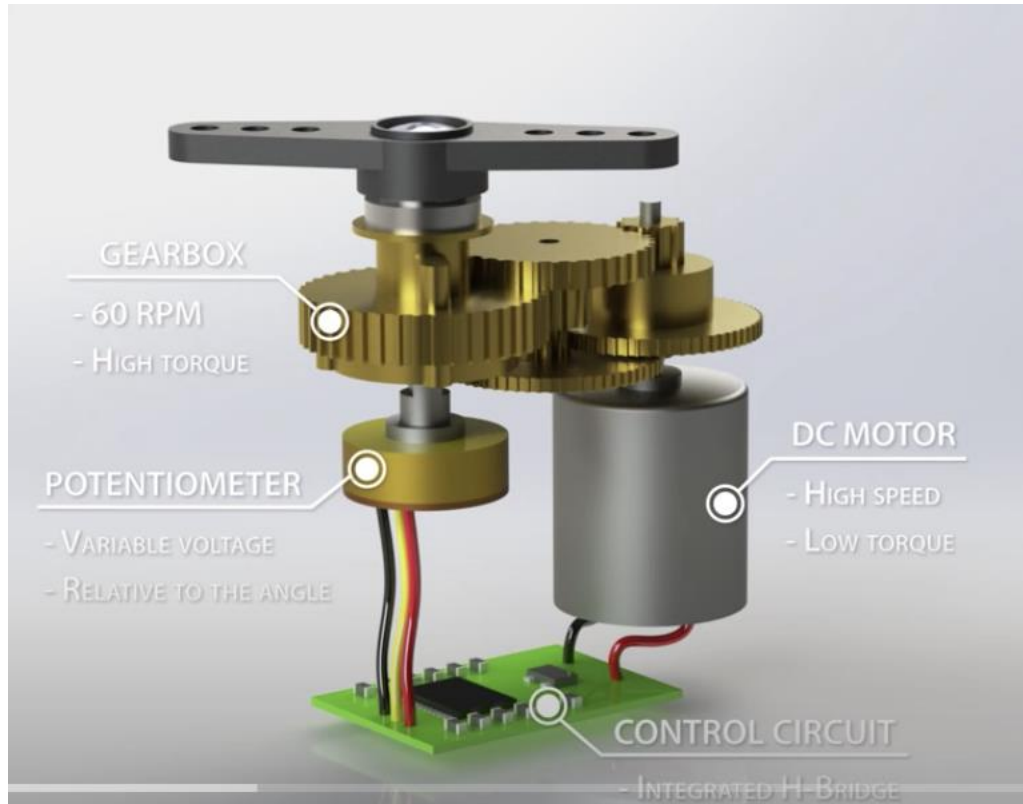
```
#include <Servo.h>

Servo myservo; // create servo object to control a servo
// twelve servo objects can be created on most boards

int pos = 0;    // variable to store the servo position

void setup() {
  myservo.attach(9); // attaches the servo on pin 9 to the servo object
}

void loop() {
  for (pos = 0; pos <= 180; pos += 1) { // goes from 0 degrees to 180 degrees
    // in steps of 1 degree
    myservo.write(pos);                // tell servo to go to position in variable 'pos'
    delay(15);                          // waits 15ms for the servo to reach the position
  }
  for (pos = 180; pos >= 0; pos -= 1) { // goes from 180 degrees to 0 degrees
    myservo.write(pos);                // tell servo to go to position in variable 'pos'
    delay(15);                          // waits 15ms for the servo to reach the position
  }
}
```



4 Stepper motor

Why stepper motors?

The rotation of stepper motors is incremental, slow and precise, while DC motors have a fast, continuous motion.

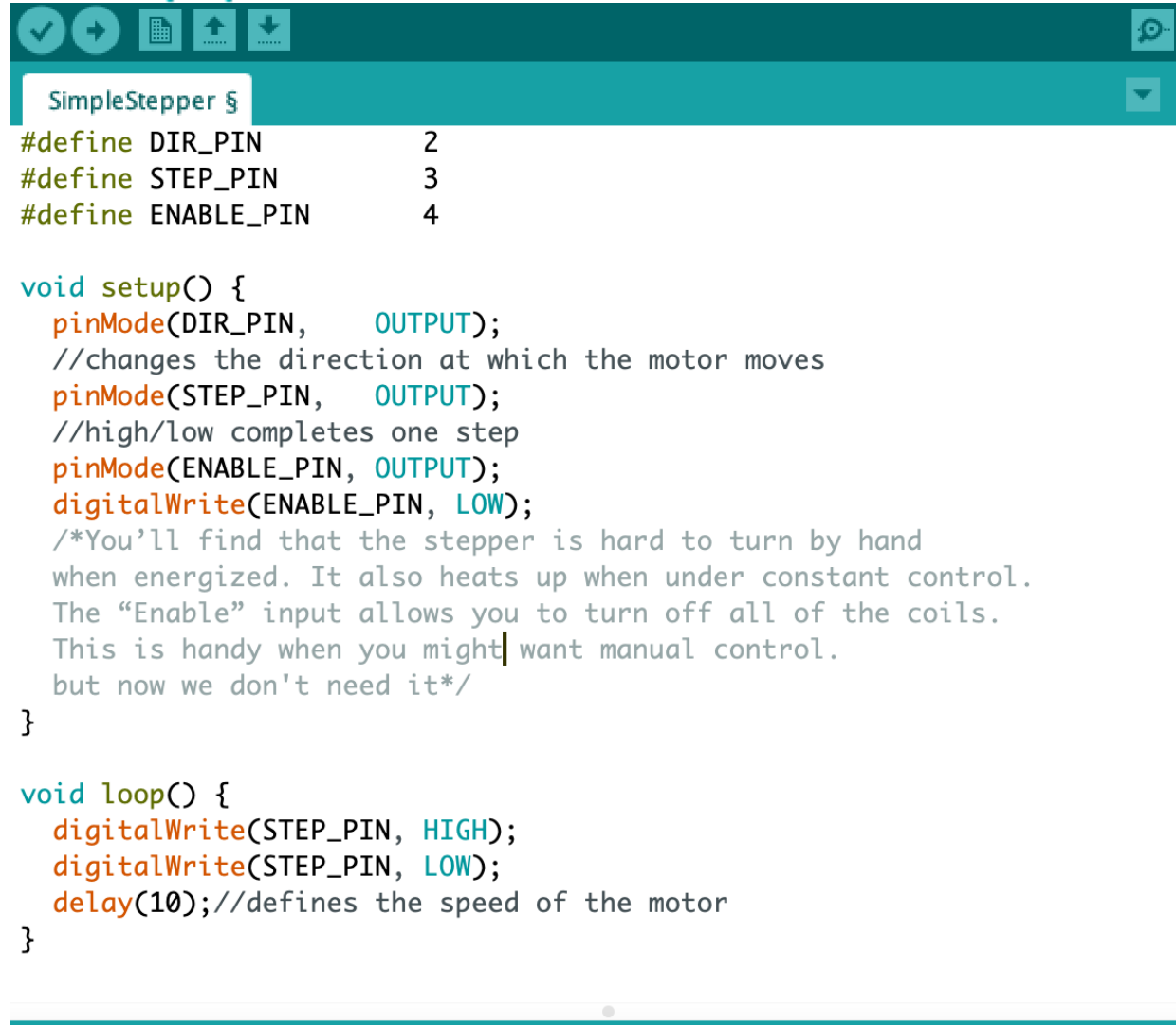
Stepper motors can move more accurately and precisely than the servo motor and are much easier to control.

Stepper motors can be easily controlled with microprocessors like the Arduino

Play with 28BYJ48 stepper motor

- <https://lastminuteengineers.com/28byj48-stepper-motor-arduino-tutorial/>

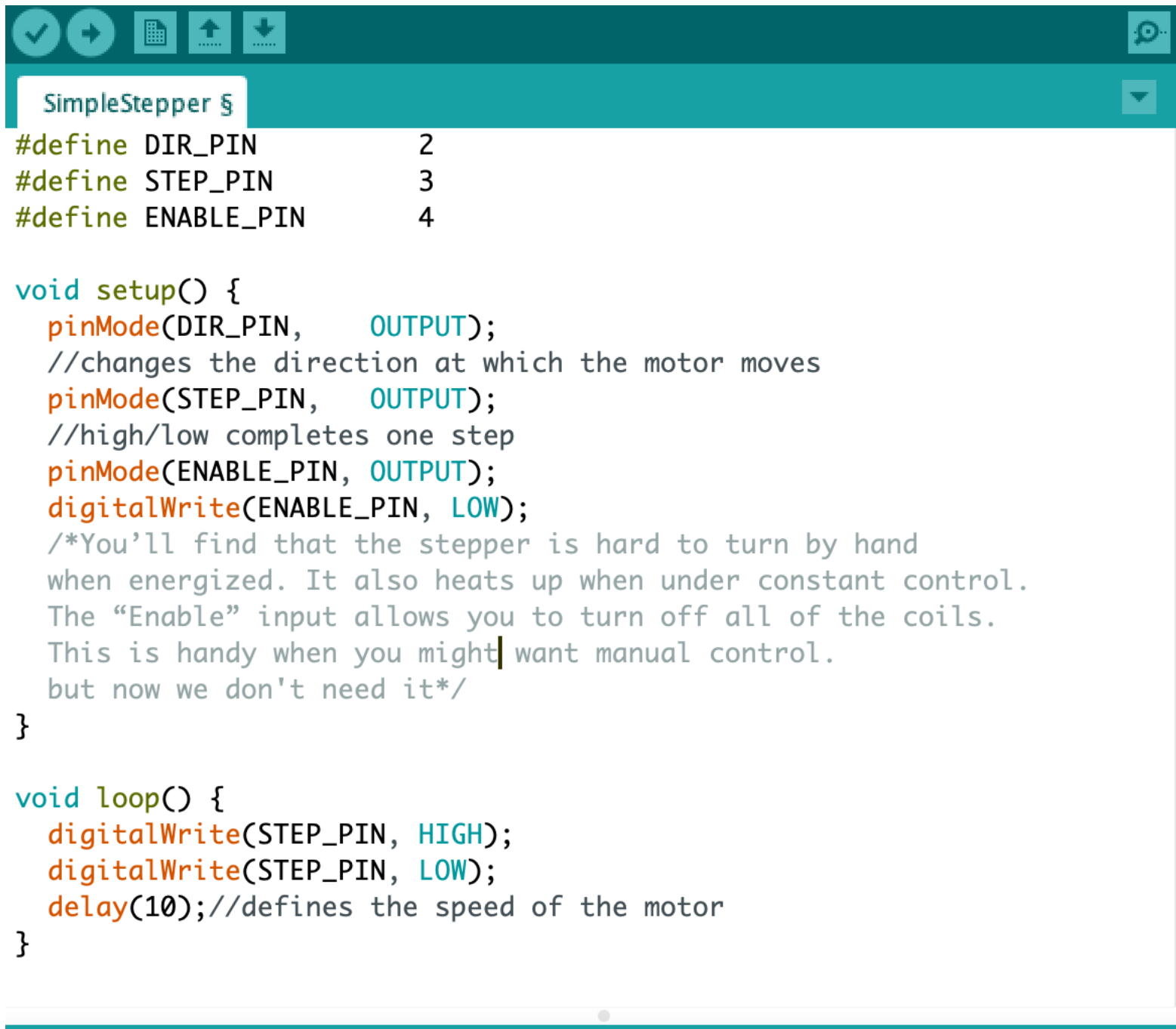
4 Stepper motor NEMA



```
SimpleStepper $
#define DIR_PIN      2
#define STEP_PIN     3
#define ENABLE_PIN   4

void setup() {
  pinMode(DIR_PIN,    OUTPUT);
  //changes the direction at which the motor moves
  pinMode(STEP_PIN,   OUTPUT);
  //high/low completes one step
  pinMode(ENABLE_PIN, OUTPUT);
  digitalWrite(ENABLE_PIN, LOW);
  /*You'll find that the stepper is hard to turn by hand
  when energized. It also heats up when under constant control.
  The "Enable" input allows you to turn off all of the coils.
  This is handy when you might want manual control.
  but now we don't need it*/
}

void loop() {
  digitalWrite(STEP_PIN, HIGH);
  digitalWrite(STEP_PIN, LOW);
  delay(10); //defines the speed of the motor
}
```



```
#define DIR_PIN      2
#define STEP_PIN     3
#define ENABLE_PIN   4

void setup() {
  pinMode(DIR_PIN,    OUTPUT);
  //changes the direction at which the motor moves
  pinMode(STEP_PIN,   OUTPUT);
  //high/low completes one step
  pinMode(ENABLE_PIN, OUTPUT);
  digitalWrite(ENABLE_PIN, LOW);
  /*You'll find that the stepper is hard to turn by hand
  when energized. It also heats up when under constant control.
  The "Enable" input allows you to turn off all of the coils.
  This is handy when you might want manual control.
  but now we don't need it*/
}

void loop() {
  digitalWrite(STEP_PIN, HIGH);
  digitalWrite(STEP_PIN, LOW);
  delay(10); //defines the speed of the motor
}
```

5 Now we move on to 6 motors!

- git clone
<https://github.com/OxRAMSociety/RobotArm/blob/main/Electronics/setup-tests/6motors/6motors.ino>

