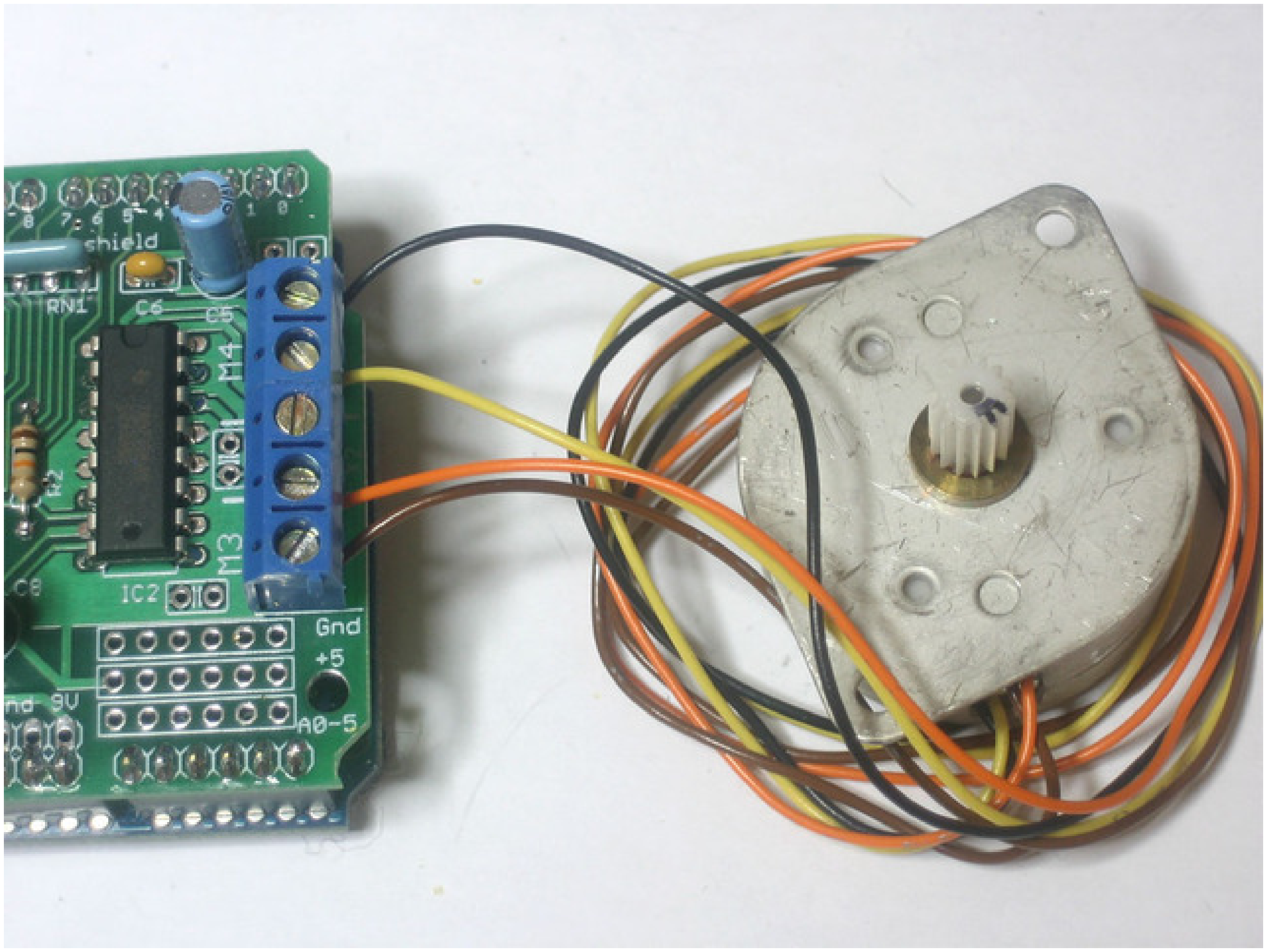
Using Stepper Motors

This tutorial is for the now ancient V1 Motor shield. Chances are you have a V2, check out the tutorial https://learn.adafruit.com/adafruit-motor-shield-v2-for-arduino This tutorial is for historical reference and previous customers only!



Stepper motors are great for (semi-)precise control, perfect for many robot and CNC projects. This motor shield supports up to 2 stepper motors. The library works identically for bi-polar and uni-polar motors

For unipolar motors: to connect up the stepper, first figure out which pins connected to which coil, and which pins are the center taps. If its a 5-wire motor then there will be 1 that is the center tap for both coils. Theres plenty of tutorials online on how to reverse engineer the coils pinout. (http://adafru.it/aOO) The center taps should both be connected together to the GND terminal on the motor shield output block. then coil 1 should connect to one motor port (say M1 or M3) and coil 2 should connect to the other motor port (M2 or M4).

For bipolar motors: its just like unipolar motors except theres no 5th wire to connect to ground. The code is exactly the same.

Running a stepper is a little more intricate than running a DC motor but its still very easy

1. Make sure you #include <AFMotor.h>
2. Create the stepper motor object with **AF_Stepper(*steps*, *stepper#*)** to setup the motor H-bridge and latches. ***Steps*** indicates how many steps per revolution the motor has. a 7.5degree/step motor has 360/7.5 = 48 steps. ***Stepper#*** is which port it is connected to. If you're using M1 and M2, its port 1. If you're using M3 and M4 it's port 2
3. Set the speed of the motor using **setSpeed(*rpm*)** where ***rpm*** is how many revolutions per minute you want the stepper to turn.
4. Then every time you want the motor to move, call the **step(*#steps*, *direction*, *steptype*)** procedure.***#steps*** is how many steps you'd like it to take**.** ***direction*** is either **FORWARD** or **BACKWARD** and the step type is **SINGLE, DOUBLE. INTERLEAVE** or **MICROSTEP**. "Single" means single-coil activation, "double" means 2 coils are activated at once (for higher torque) and "interleave" means that it alternates between single and double to get twice the resolution (but of course its half the speed). "Microstepping" is a method where the coils are PWM'd to create smooth motion between steps. Theres tons of information about the pros and cons of these different stepping methods in the resources page. (http://adafru.it/aOO) You can use whichever stepping method you want, changing it "on the fly" to as you may want minimum power, more torque, or more precision.
5. By default, the motor will 'hold' the position after its done stepping. If you want to release all the coils, so that it can spin freely, call **release()**
6. The stepping commands are 'blocking' and will return once the steps have finished.

Because the stepping commands 'block' - you have to instruct the Stepper motors each time you want them to move. If you want to have more of a 'background task' stepper control, check out AccelStepper library (http://adafru.it/aOL) (install similarly to how you did with AFMotor) which has some examples for controlling two steppers simultaneously with varying accelleration

```
#include <AFMotor.h>


AF_Stepper motor(48, 2);


void setup() {
  Serial.begin(9600);           // set up Serial library at 9600 bps
  Serial.println("Stepper test!");

  motor.setSpeed(10);  // 10 rpm

  motor.step(100, FORWARD, SINGLE);
  motor.release();
  delay(1000);
}

void loop() {
  motor.step(100, FORWARD, SINGLE);
  motor.step(100, BACKWARD, SINGLE);

  motor.step(100, FORWARD, DOUBLE);
  motor.step(100, BACKWARD, DOUBLE);

  motor.step(100, FORWARD, INTERLEAVE);
  motor.step(100, BACKWARD, INTERLEAVE);

  motor.step(100, FORWARD, MICROSTEP);
  motor.step(100, BACKWARD, MICROSTEP);
}
```