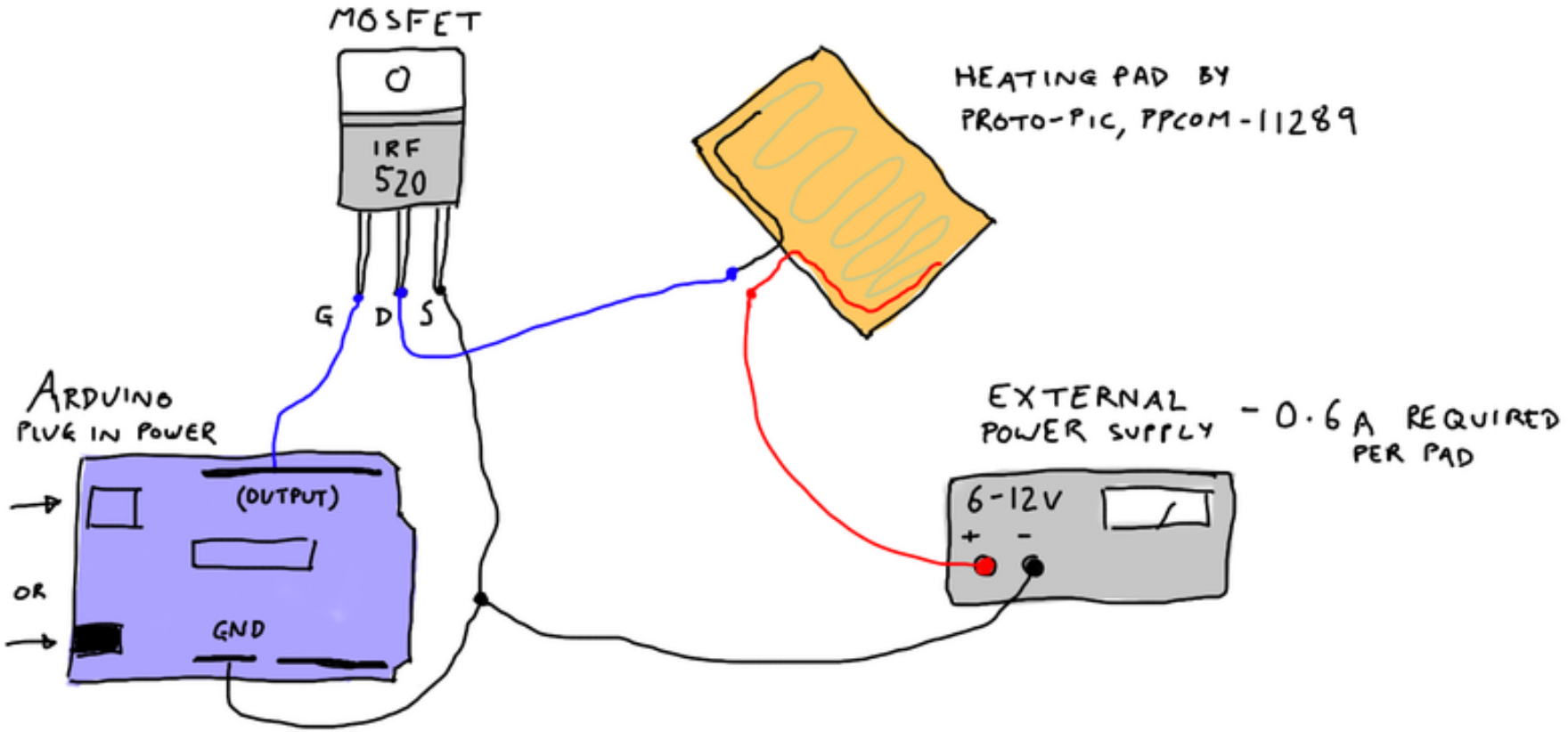
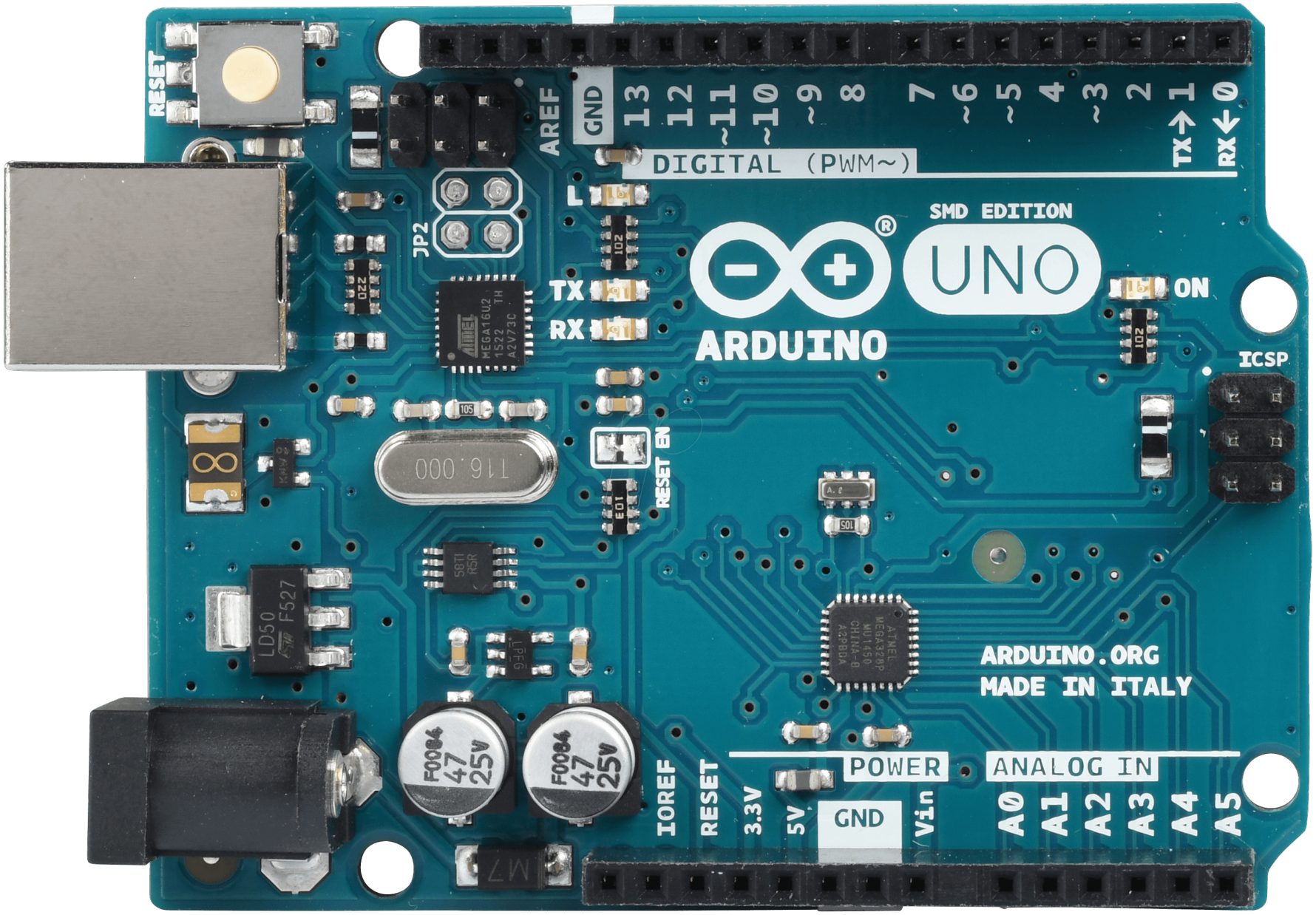

MOSFET
IRF
520
G
D
S
HEATING PAD BY
PROTO-PIC, PPCOM-11289
ARDUINO
PLUG IN POWER
(OUTPUT)
OR
GND
EXTERNAL
POWER SUPPLY
6-12V
+
-
- 0.6A REQUIRED
PER PAD

# of using microcontroller with Heater Circuit

- **Problem**
  - Circuit draws more current than available from the Arduino
- **Solution**
  - Use external Power (9v)
  - transistor (MOSFET). Works as a stitch
  - They can be controlled by a low current source (like a microcontroller) but can connect a higher current source (like a battery or DC supply)

Introduction to the Arduino Board


RESET
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM~)
SMD EDITION
UNO
ARDUINO
ON
ICSP
JP2
TX
RX
RESET EN
ARDUINO.ORG
MADE IN ITALY
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
Vin
A0
A1
A2
A3
A4
A5

Get going

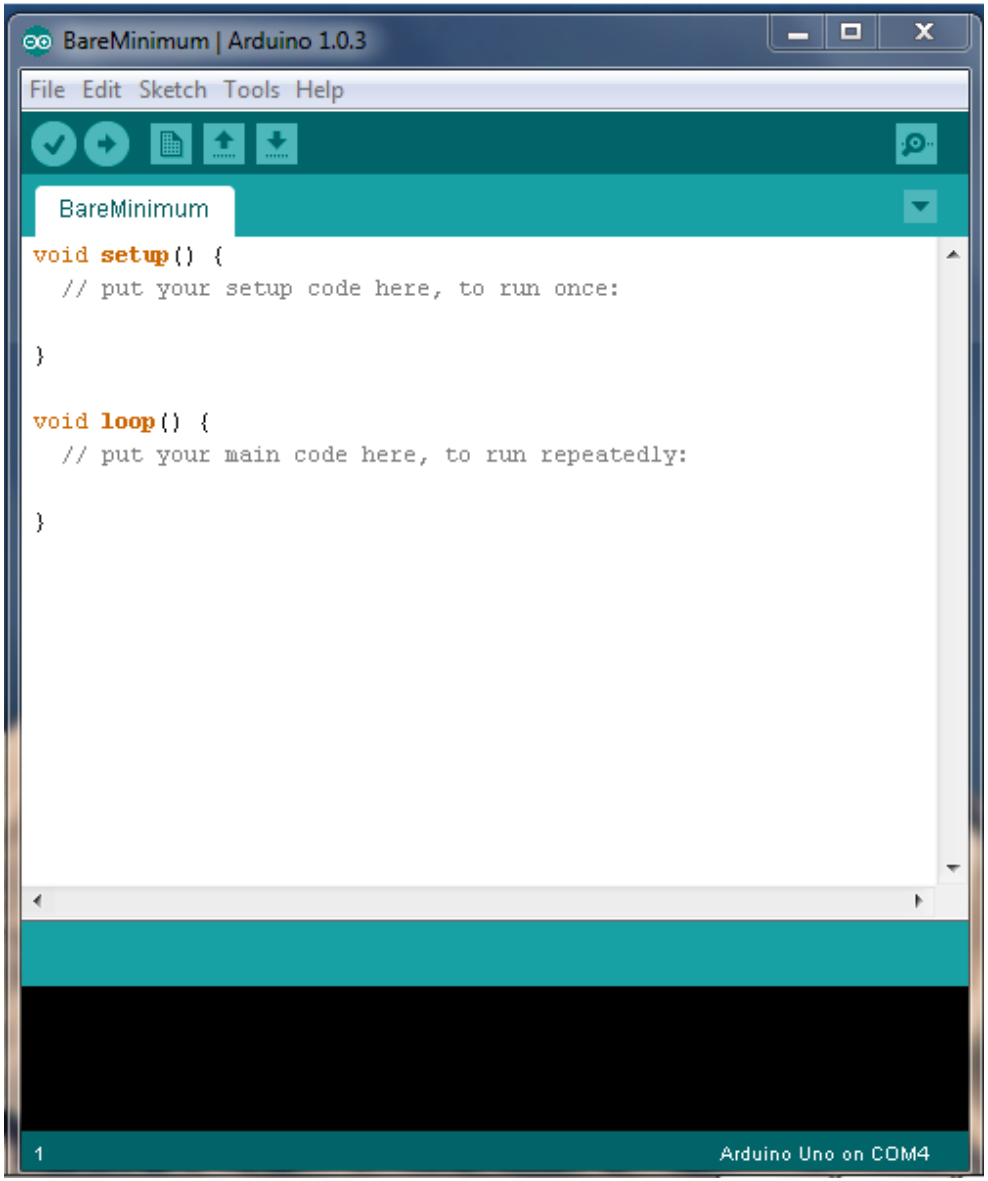
- In the IDE goto:

  Tools > Board > Arduino/Genuino Uno

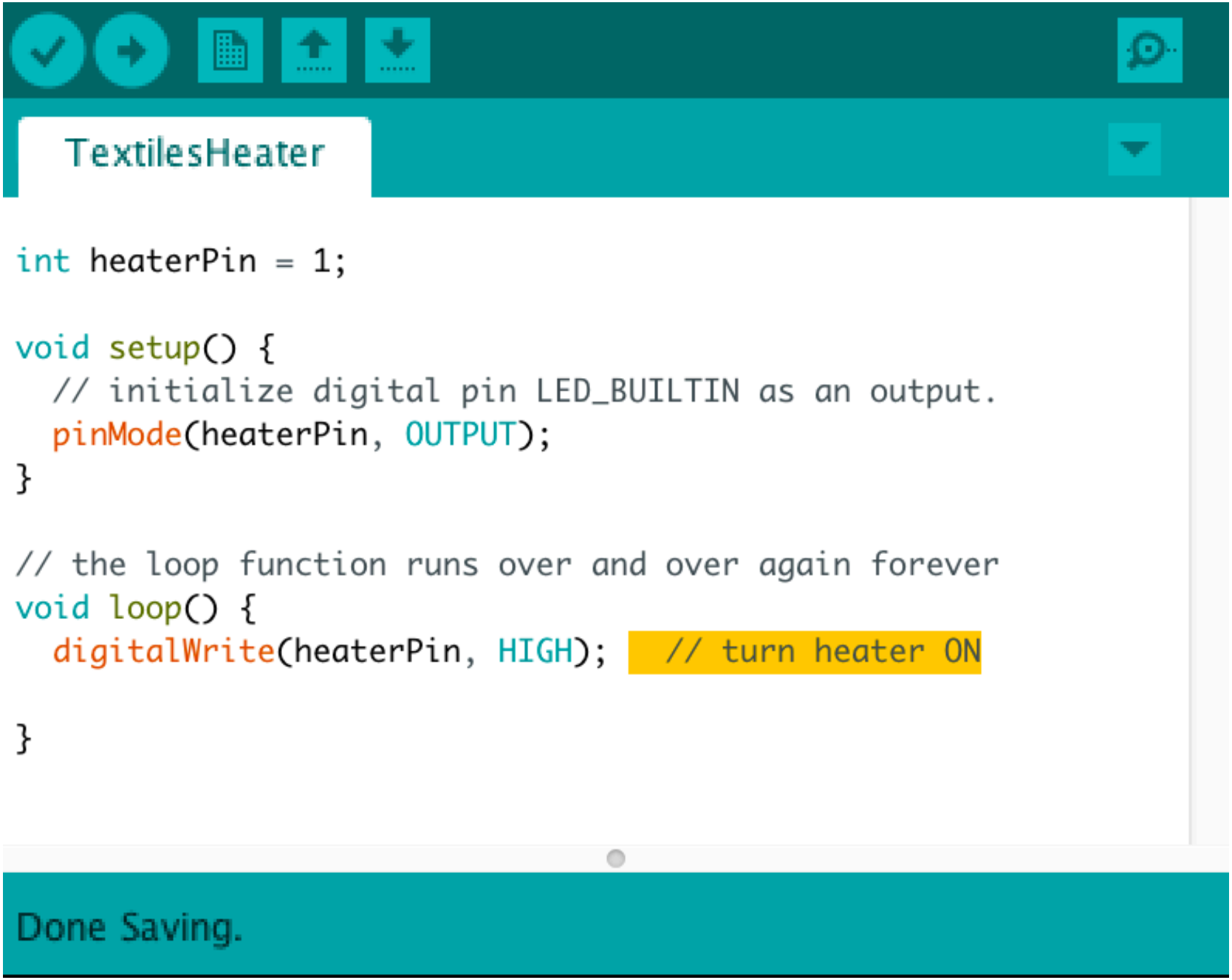
  Tools > Port > COM[!] Arduino/Genuino Uno

Arduino IDE

- integrated development environment



https://raw.githubusercontent.com/JohnMechatronics/Wearable_Electronics/master/TextilesHeater.ino


TextilesHeater
int heaterPin = 1;
void setup() {
// initialize digital pin LED_BUILTIN as an output.
pinMode(heaterPin, OUTPUT);
}
// the loop function runs over and over again forever
void loop() {
digitalWrite(heaterPin, HIGH); // turn heater ON
}
Done Saving.

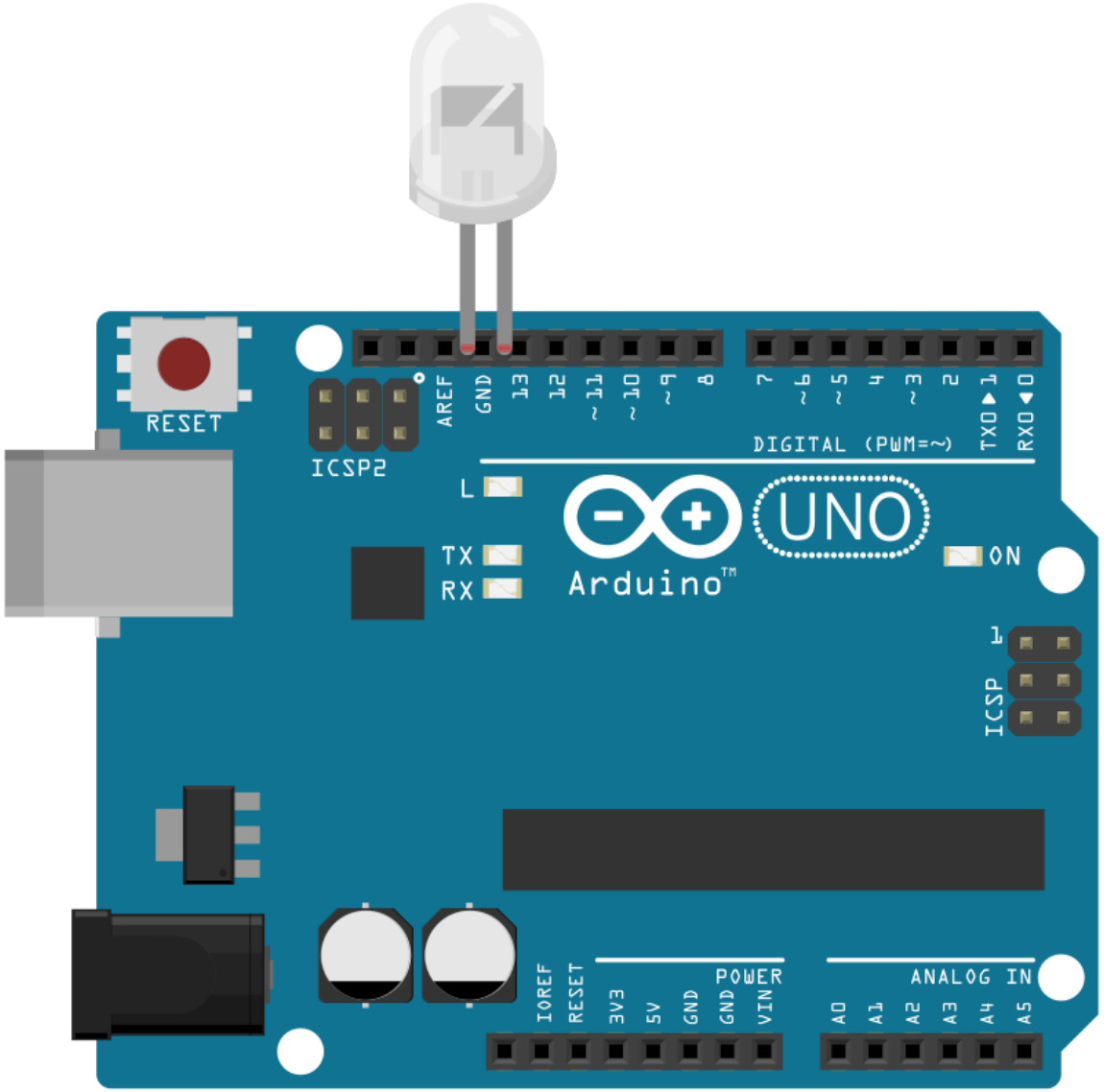
digitalWrite()

If the pin has been configured as
an OUTPUT with pinMode(), its voltage will be set to:

- 5V for HIGH
- 0V (ground) for LOW.

File > Examples > 0.1 Basics > Blink

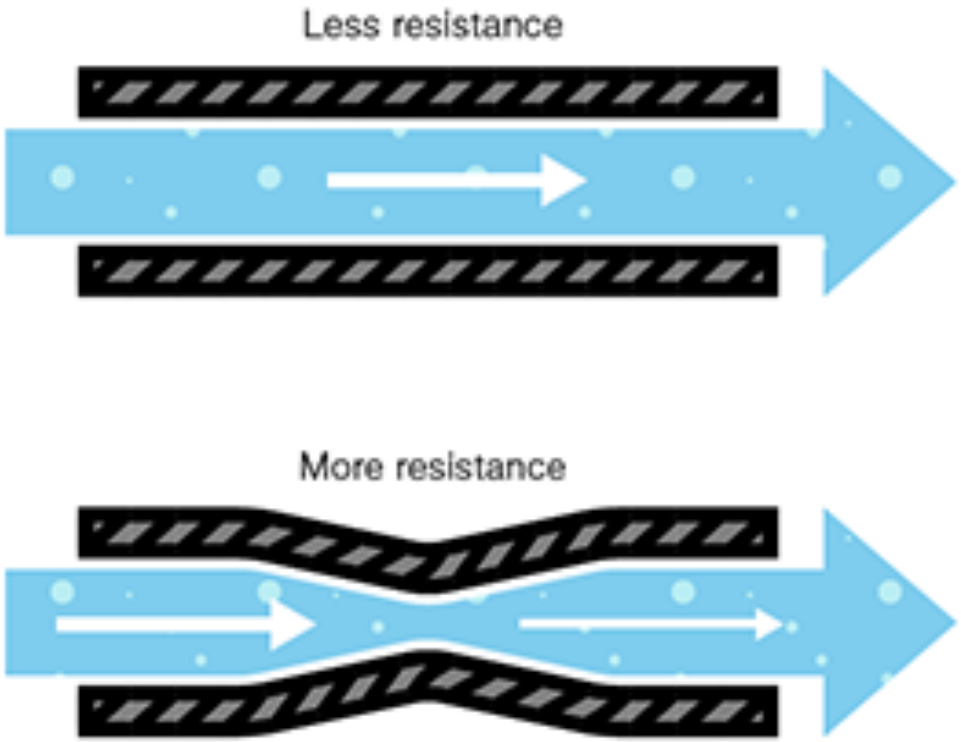

(anode)
+
(cathode)
GND


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
Arduino™
UNO
ON
ICSP
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN

Basic Electronics

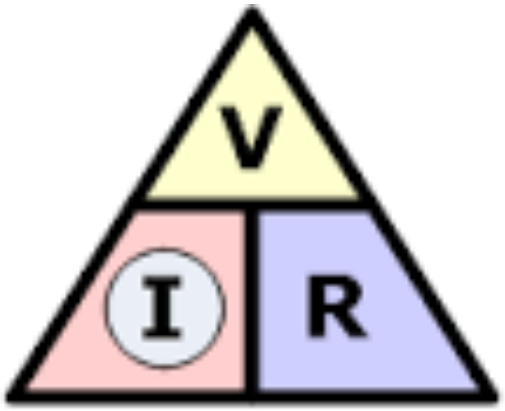
- **Voltage** – The difference in electrical energy between two points. It is measured in volts (**v**).
- **Current** – The quantity or amount of electrical energy passing a particular point. It is measured in amps (**A**)
- **Resistance** - The measure of a materials ability to prevent the flow of electricity. Resistance is measured in ohms (**Ω**)

Resistance

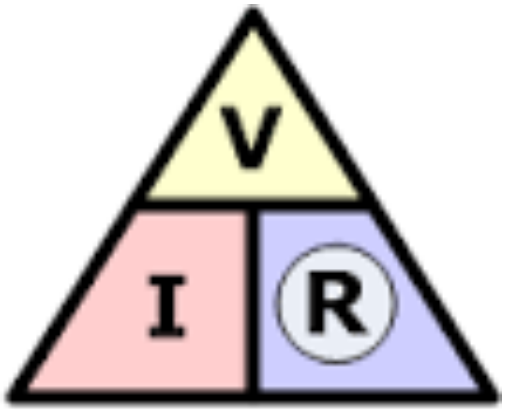


Ohm's law

$$V = I \times R$$



$$I = \frac{V}{R}$$



$$R = \frac{V}{I}$$

Ohm's law states that **Voltage** (V) is equal to **current** (A) times **resistance** ()

Work Out the Resistance

$$R = (Vs - Vf)/I$$

Vs = Source Voltage

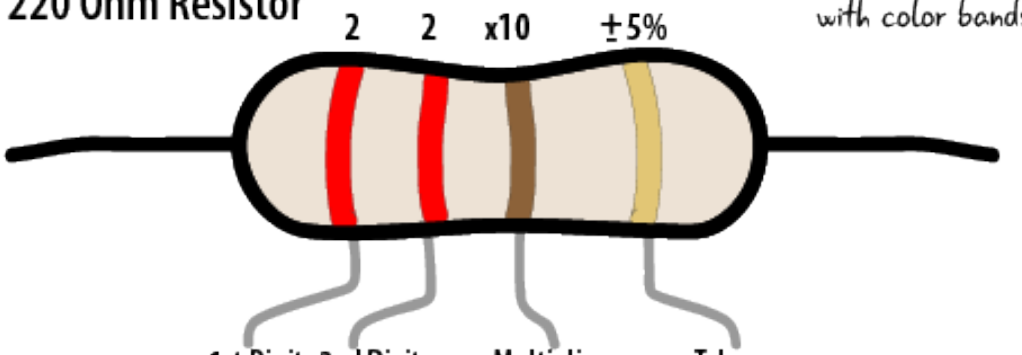
Vf = LED Forward Voltage

I = LED required voltage

220 Ohm Resistor

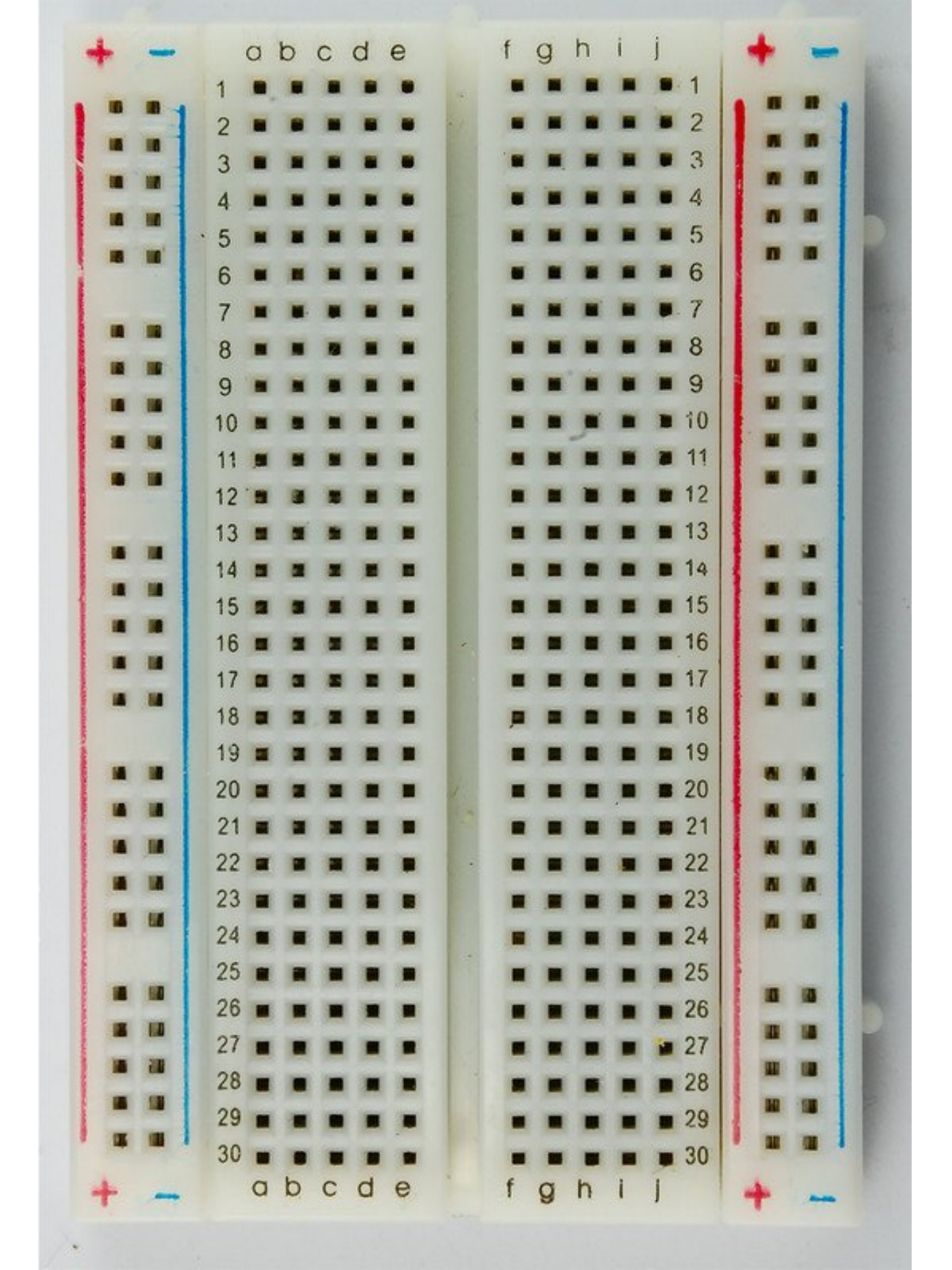
220 Ohm Resistor

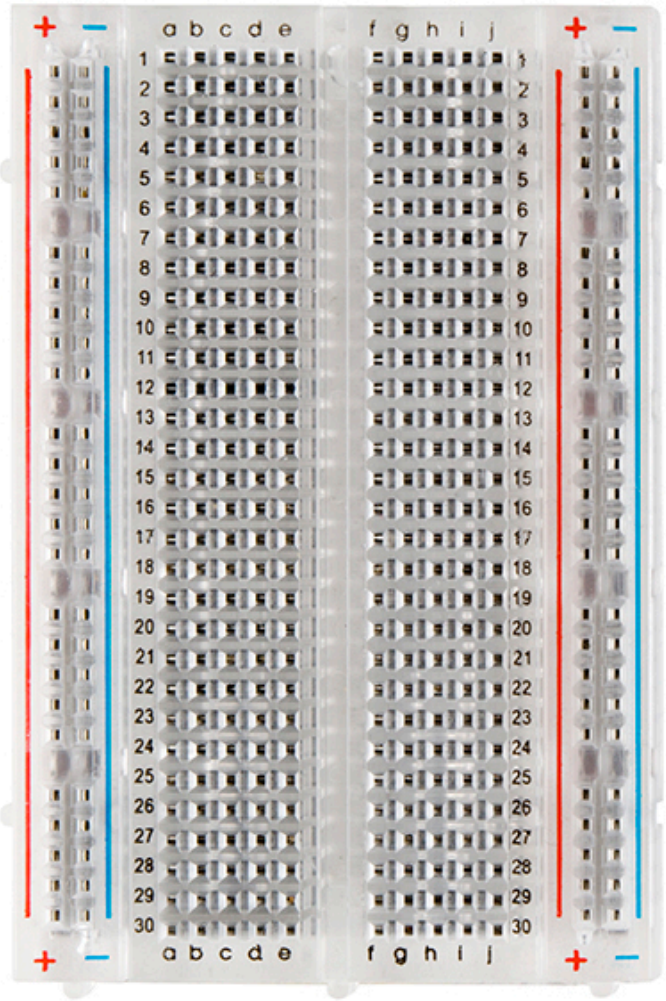
sample resistor with color bands

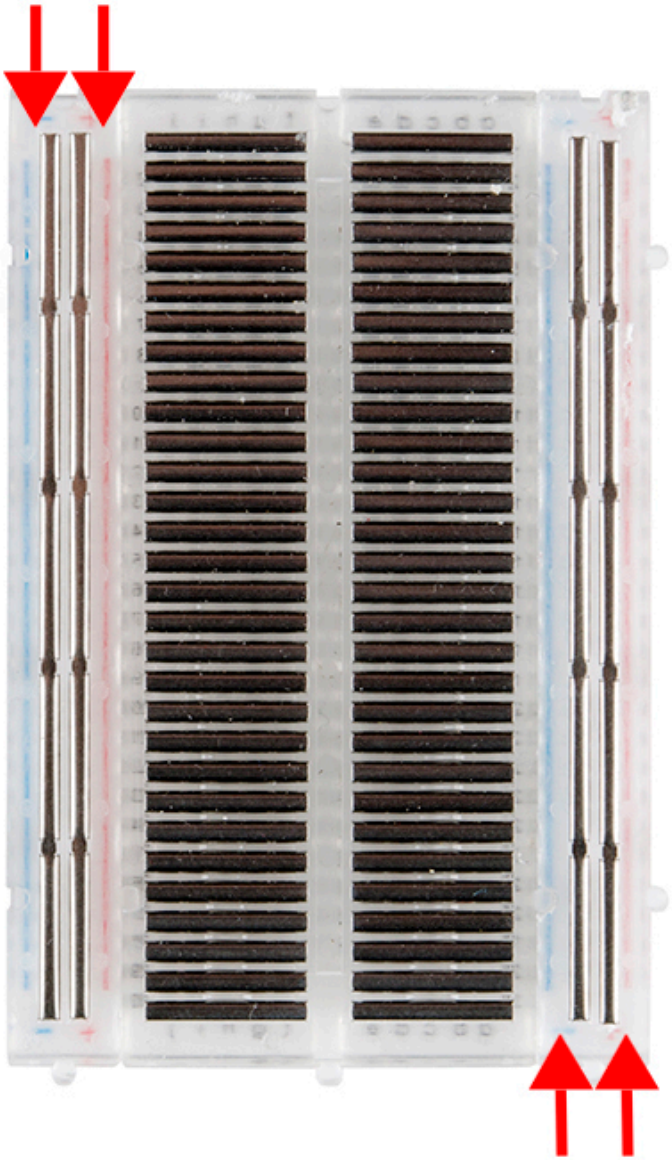


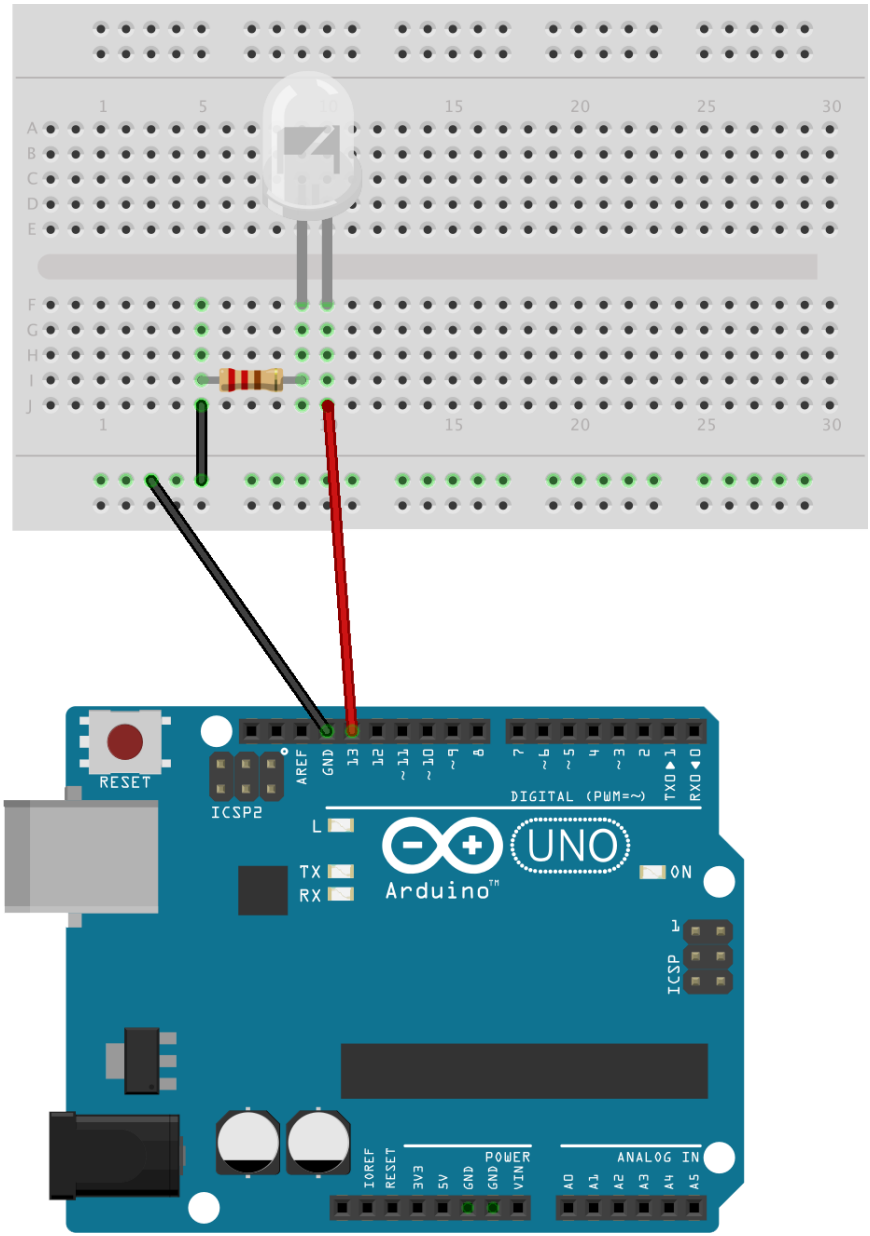
what the color bands represent on the resistor

	1st Digit	2nd Digit	Multiplier	Tolerance
Black	0	0	1	5% Gold
Brown	1	1	10	10% Silver
Red	2	2	100	
Orange	3	3	1,000	
Yellow	4	4	10,000	
Green	5	5	100,000	
Blue	6	6	1,000,000	
Purple	7	7		
Gray	8	8		
White	9	9		


a b c d e
f g h i j
1 2 3 4 5 6 7 8 9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28 29 30
a b c d e
f g h i j


a b c d e
f g h i j
1
2
3
4
5
6
7
8
9
10
11
12
13
14
15
16
17
18
19
20
21
22
23
24
25
26
27
28
29
30




1
5
10
15
20
25
30
A
B
C
D
E
F
G
H
I
J
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

```
int ledPin = 13;    // LED connected to digital pin 13

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

// the loop function runs over and over again forever
void loop() {
  digitalWrite(ledPin, HIGH);   // turn the LED on (HIGH is the voltage level)
  delay(1000);                  // wait for a second
  digitalWrite(ledPin, LOW);    // turn the LED off by making the voltage LOW
  delay(1000);                  // wait for a second
}
```

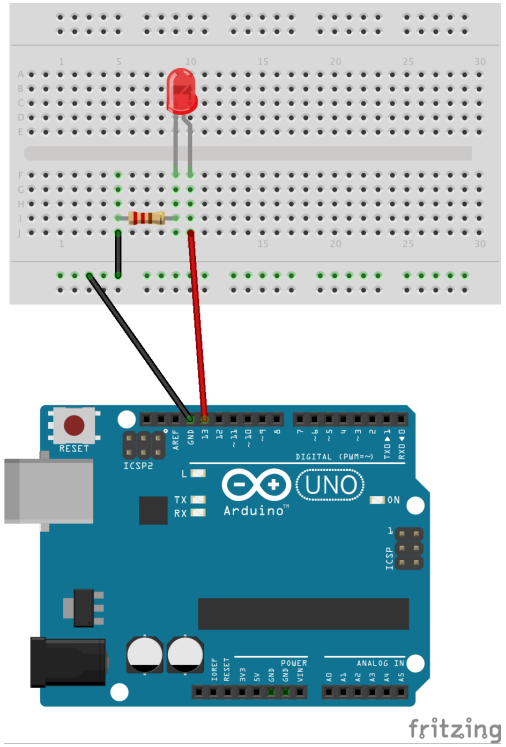
Variables

- A variable is a way of naming and storing a value for use by the program
- E.g. data from a sensor or value that will be transformed by a mathematical calculation.

```
int ledPin = 13;
digitalWrite(ledPin, HIGH);  -> digitalWrite(13, HIGH);
```

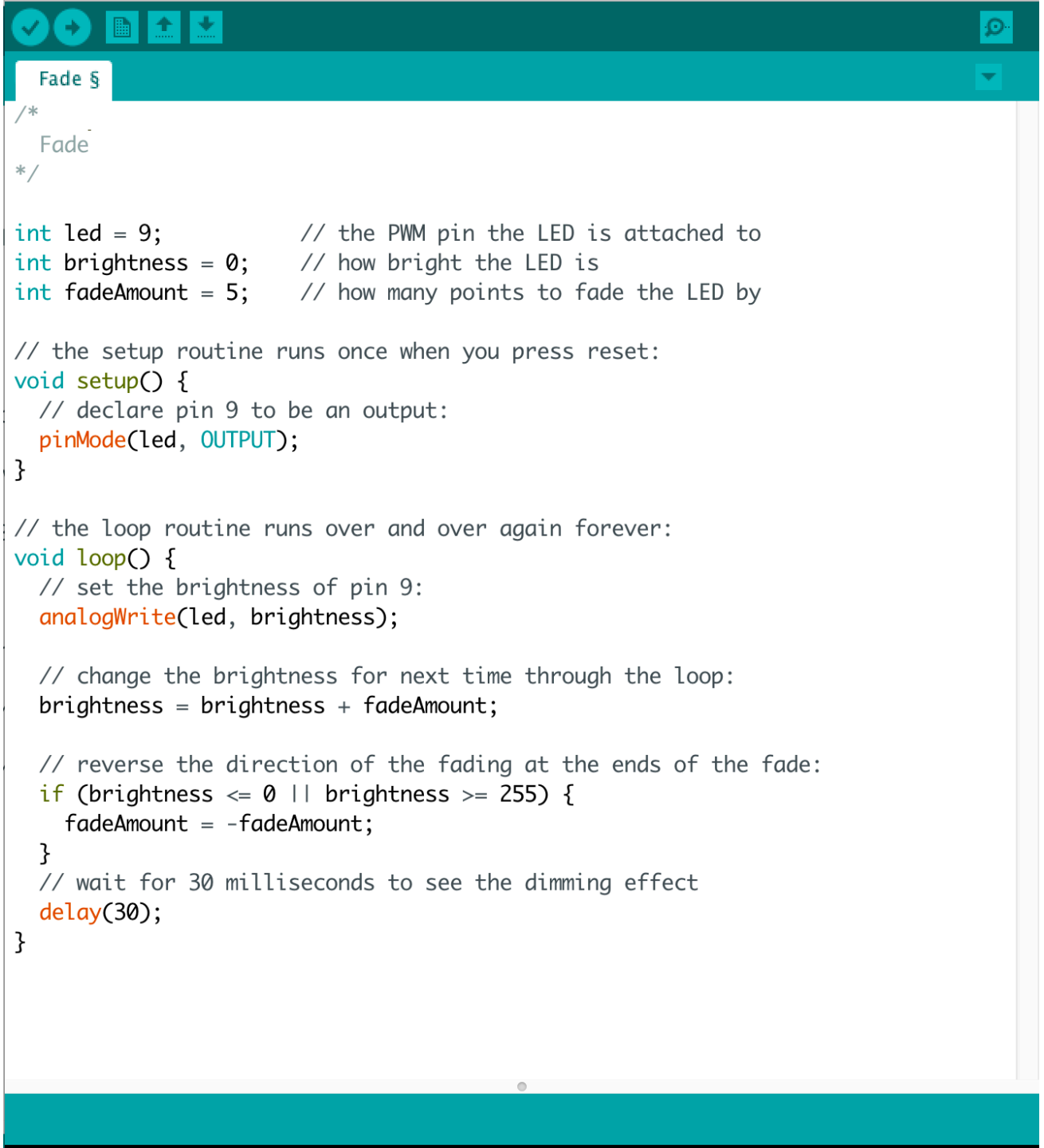
Task

- Test the circuit
- Can you alter the blink time?
- Try different rates



Fade

- Rewire the LED to pin 9
- From the Arduino IED open:
- File > Example > 01.Basics > Fade
- Upload the Code
- What does it do?


Fade §
/*
Fade
*/
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);
}

analogWrite() v digitalWrite()

digitalWrite()

- Two states High or Low
- High - > 5v
- Low - > 0v

analogWrite()

- Set the intensity between 0 and 255

if

```
if (brightness <= 0 || brightness >= 255) {
    fadeAmount = -fadeAmount;
}
```

*Remember a – and a – make a +

if

- The if() statement allows you to make something happen or not, depending on whether a given condition is true or false. Eg:

```
if (someCondition) {
    // do stuff if the condition is true
}
```

if-else

```
if (someCondition) {
    // do stuff if the condition is true
} else {
    // do stuff if the condition is false
}
```

else-if

```
if (someCondition) {
    // do stuff if the condition is true
} else if (anotherCondition) {
    // do stuff only if the first condition is false
     // and the second condition is true
}
```

Comparison Operators

!= (not equal to)
< (less than)
<= (less than or equal to)
== (equal to)
> (greater than)
>= (greater than or equal to)

```
if (brightness <= 0 ||
brightness >= 255)
```

Boolean Operators

&& - and

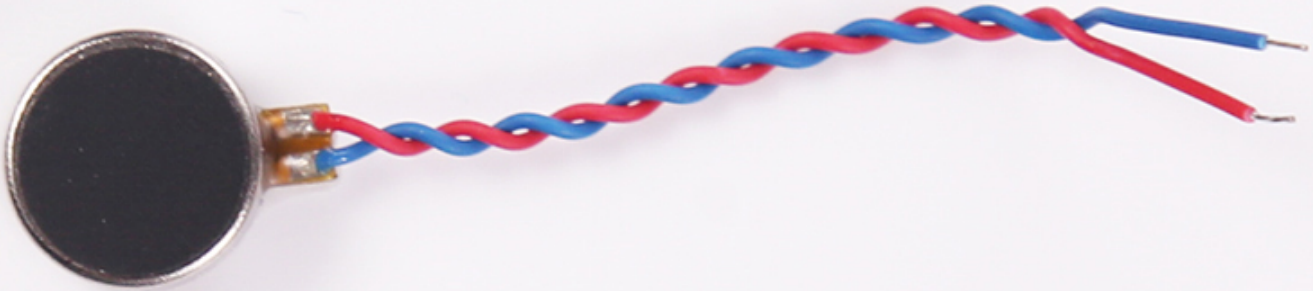
|| - or

```
if (brightness <= 0 || brightness >= 255)
```

Haptic

'Haptic or kinesthetic communication recreates the sense of touch by applying forces, vibrations, or motions to the user'.

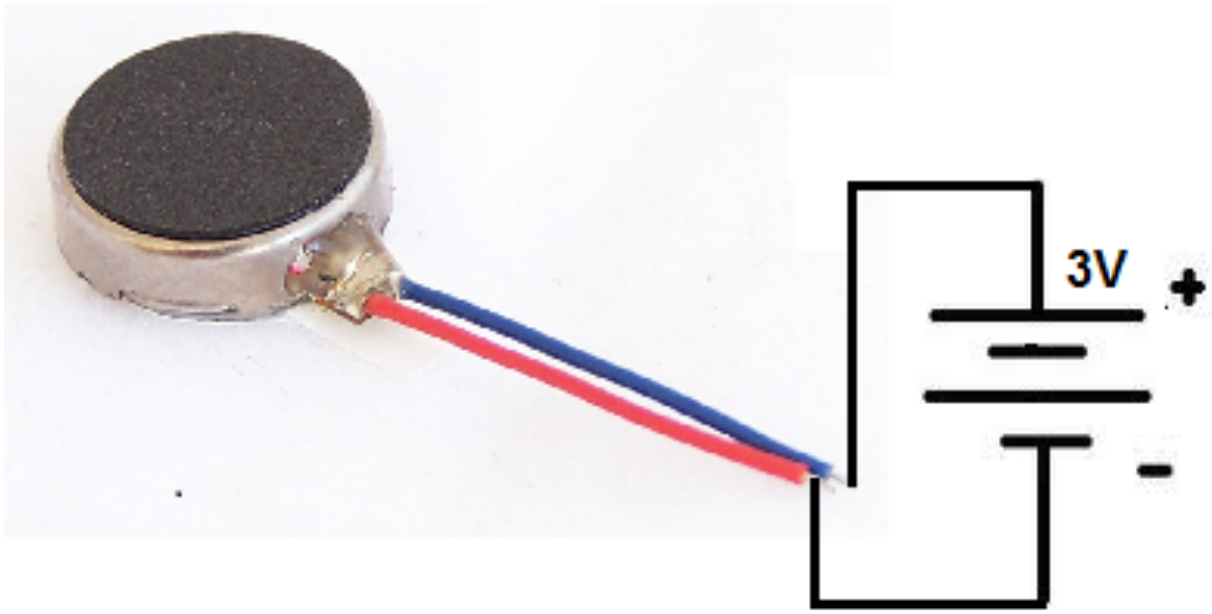
Gabriel Robles-De-La-Torre. "International Society for Haptics: Haptic technology, an animated explanation". Isfh.org. Archived from the original on 2010-03-07. Retrieved 2010-02-26



Good Vibrations!

- A vibration motor is a motor which vibrates when given sufficient power.
- It is very good for vibrating objects.
- eg. cell phone, a game controller, furby toy!

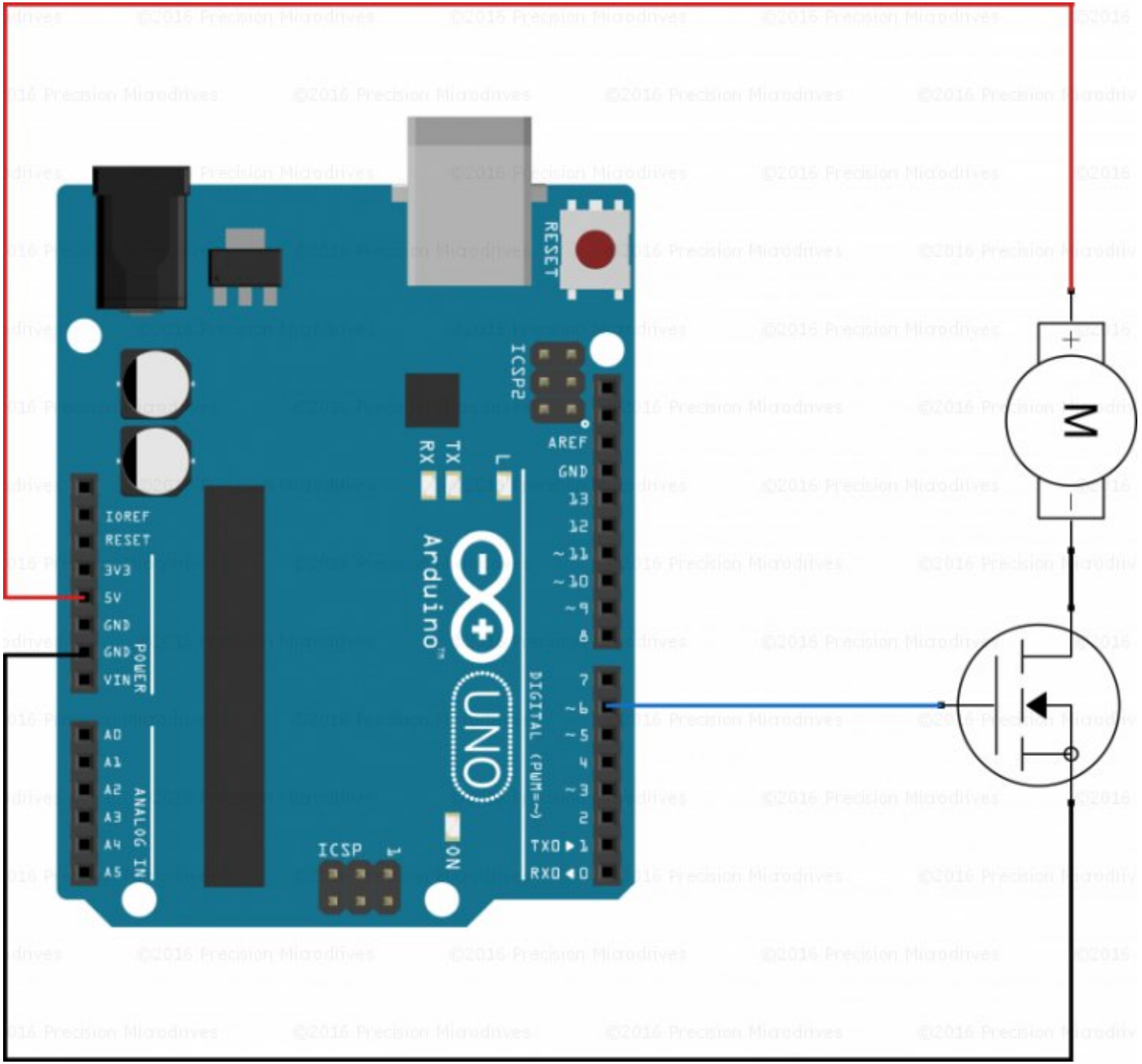
- Simple way to give Haptic feedback



If we connect 3 volts across its terminal, it will vibrate really well

Problem

- **Arduino** Pins do not have enough current to start the motor. Could blow the Arduino
- The 5v pin does have enough current.
- **Solution** use a N-type MOSFET, this time the 2N7000

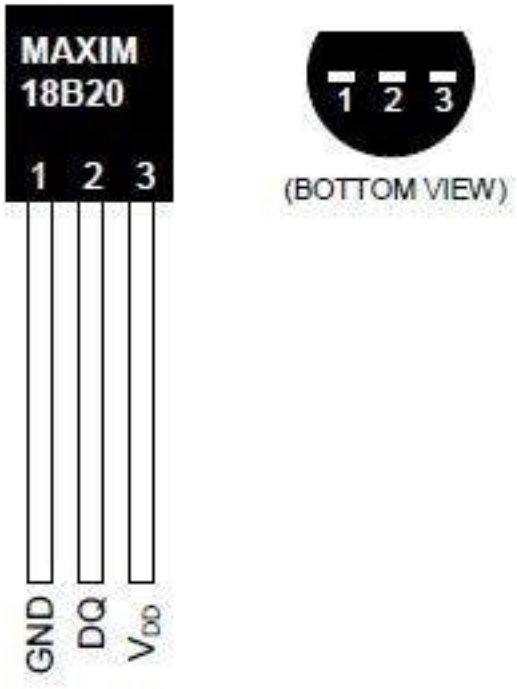

RESET
ICSP2
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
4
~3
2
TX→1
RX←0
DIGITAL (PWM=~)
TX
RX
L
Arduino
UNO
ON
ICSP
1
IOREF
RESET
3V3
5V
GND
GND
VIN
POWER
A0
A1
A2
A3
A4
A5
ANALOG IN
M

vibrationMotorPWM

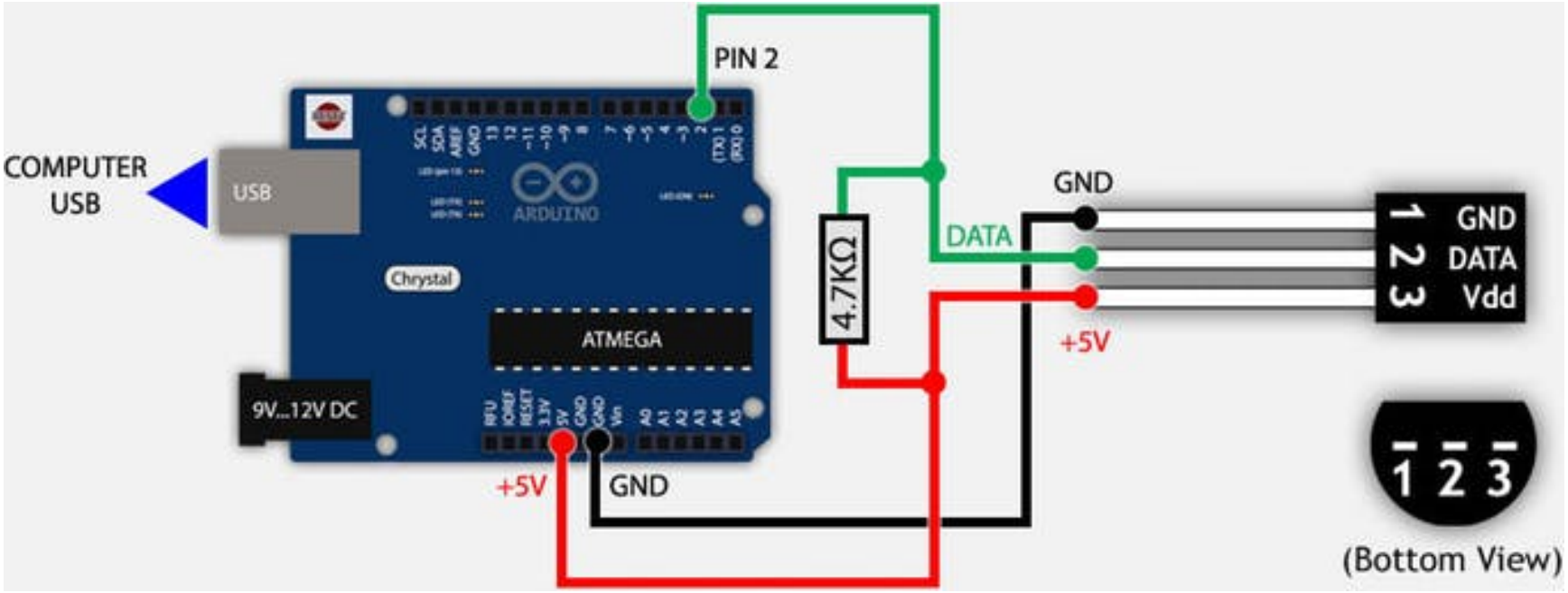
```
void setup() {
  pinMode( 6 , OUTPUT);  // Must be a PWM pin
}

void loop() {

  analogWrite( 6 , 153 );  // 60% duty cycle
  delay(500);              // play for 0.5s

  analogWrite( 6 , 0 );    // 0% duty cycle (off)
  delay(4000);             // wait for 4s

}
```

Input – Heat

DS18B20 Digital Temperature Sensor



Measures temperatures from –55°C to +125°C


PIN 2
COMPUTER
USB
USB
ARDUINO
Chrystal
ATMEGA
9V...12V DC
+5V
GND
4.7KΩ
DATA
GND
+5V
GND
DATA
Vdd
1 2 3
1 2 3
(Bottom View)

TempSensor

```
#include <OneWire.h>
#include <DallasTemperature.h>

// Data wire is plugged into pin 2 on the Arduino
#define ONE_WIRE_BUS 2

// Setup a oneWire instance to communicate with any OneWire devices
// (not just Maxim/Dallas temperature ICs)
OneWire oneWire(ONE_WIRE_BUS);

// Pass our oneWire reference to Dallas Temperature.
DallasTemperature sensors(&oneWire);

void setup(void)
{
  // start serial port
  Serial.begin(9600);
  Serial.println("Dallas Temperature IC Control Library Demo");

  // Start up the library
  sensors.begin();
}


void loop(void)
{
  // call sensors.requestTemperatures() to issue a global temperature
  // request to all devices on the bus
  Serial.print(" Requesting temperatures...");
  sensors.requestTemperatures(); // Send the command to get temperatures
  Serial.println("DONE");

  Serial.print("Temperature is: ");
  Serial.print(sensors.getTempCByIndex(0)); // Why "byIndex"?
    // You can have more than one IC on the same bus.
    // 0 refers to the first IC on the wire
    delay(1000);
}
```

```
Dture is: 17.81 Requesting temperatures...DONE
Temperature is: 17.75 Requesting temperatures...Dallas Temperature IC Control Library Demo
 Requesting temperatures...DONE
Temperature is: 17.75 Requesting temperatures...DONE
Temperature is: 17.69 Requesting temperatures...DONE
Temperature is: 17.69 Requesting temperatures...DONE
Temperature is: 17.75 Requesting temperatures...DONE
Temperature is: 17.81 Requesting temperatures...DONE
Temperature is: 17.81 Requesting temperatures...DONE
Temperature is: 17.81 Requesting temperatures...DONE
Temperature is: 17.81 Requesting temperatures...DONE
Temperature is: 17.81 Requesting temperatures...DONE
Temperature is: 17.81 Requesting temperatures...DONE
Temperature is: 17.81 Requesting temperatures...DONE
Temperature is: 17.81 Requesting temperatures...DONE
Temperature is: 17.75 Requesting temperatures...DONE
Temperature is: 17.62 Requesting temperatures...DONE
Temperature is: 17.62 Requesting temperatures...DONE
Temperature is: 17.62 Requesting temperatures...DONE
Temperature is: 17.69 Requesting temperatures...DONE
Temperature is: 17.69 Requesting temperatures...DONE
Temperature is: 17.75 Requesting temperatures...DONE
Temperature is: 17.75 Requesting temperatures...DONE
Temperature is: 17.62 Requesting temperatures...DONE
Temperature is: 17.50 Requesting temperatures...DONE
Temperature is: 17.44 Requesting temperatures...DONE
Temperature is: 17.37 Requesting temperatures...DONE
Temperature is: 17.37 Requesting temperatures...DONE
Temperature is: 17.37 Requesting temperatures...DONE
Temperature is: 17.44 Requesting temperatures...DONE
Temperature is: 17.44 Requesting temperatures...DONE
Temperature is: 17.37 Requesting temperatures...DONE
Temperature is: 17.31 Requesting temperatures...DONE
Temperature is: 17.31 Requesting temperatures...DONE
Temperature is: 17.19 Requesting temperatures...DONE
Temperature is: 17.12 Requesting temperatures...DONE
Temperature is: 17.12 Requesting temperatures...DONE
```

TempSensorIF

```
#include <OneWire.h>
#include <DallasTemperature.h>

#define ONE_WIRE_BUS 2
OneWire oneWire(ONE_WIRE_BUS);
DallasTemperature sensors(&oneWire);

void setup(void)
{
  // start serial port
  Serial.begin(9600);
  Serial.println("Dallas Temperature IC Control Library Demo");

  // Start up the library
  sensors.begin();
}

void loop(void)
{
  // call sensors.requestTemperatures() to issue a global temperature
  // request to all devices on the bus

  sensors.requestTemperatures(); // Send the command to get temperatures
  int Temp = sensors.getTempCByIndex(0);// Why "byIndex"?
    // You can have more than one IC on the same bus.
    // 0 refers to the first IC on the wire

  if (Temp >0 && Temp < 15){
    Serial.print(Temp);
    Serial.println(" - is greater than 0 and less than 15");
    } else if (Temp >14 && Temp < 25){
     Serial.println(Temp);
    Serial.println(" - is greater than 14 and less than 25");
    } else if (Temp >24 && Temp < 35){
    Serial.print(Temp);
    Serial.println(" - is greater than 24 and less than 35");
    } else {
      Serial.println(Temp + " is out side of the range");
    }

    delay(50);
}
```

Done Saving.

TempSensorIF.ino

TASK!

- Add the heat sensor into your heat pad.
- Use the heat sensor results to turn the pad on and off