# Intro to Physical Computing

***aka How to make Daft Punk Helmets 101.***

## CC Lab (Fall 2016)

# Git

git pull

More…..

http://rogerdudler.github.io/git-guide/

# Terminal for PowerUser

Zsh?!

Oh-My-Zsh?!

Iterm!?

# What’s an “Arduino”?

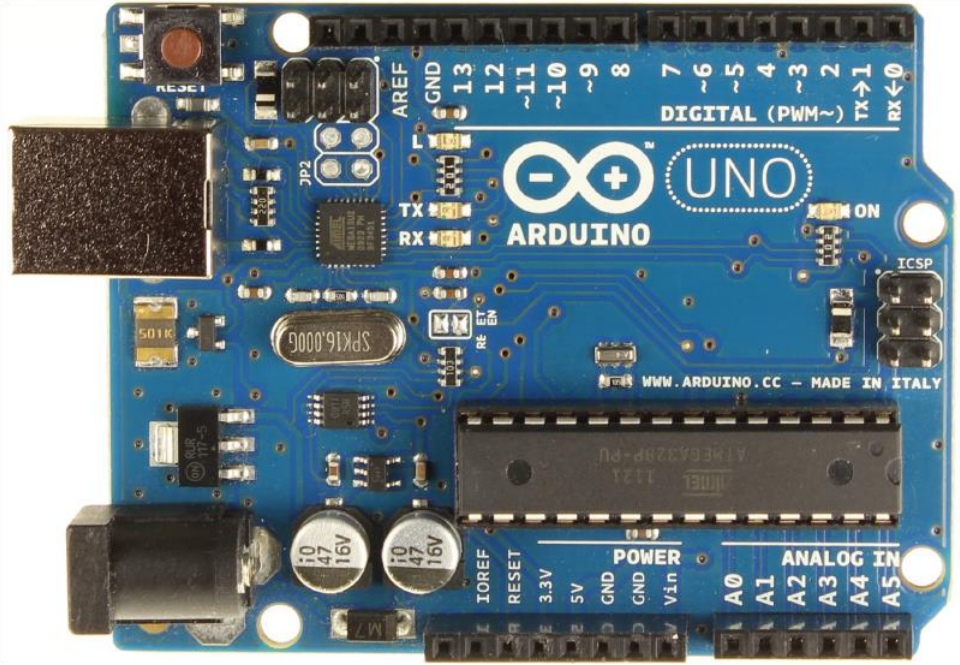
Microcontroller
- Input/Output machine

Made for rapid prototyping
- (without requiring custom boards design)

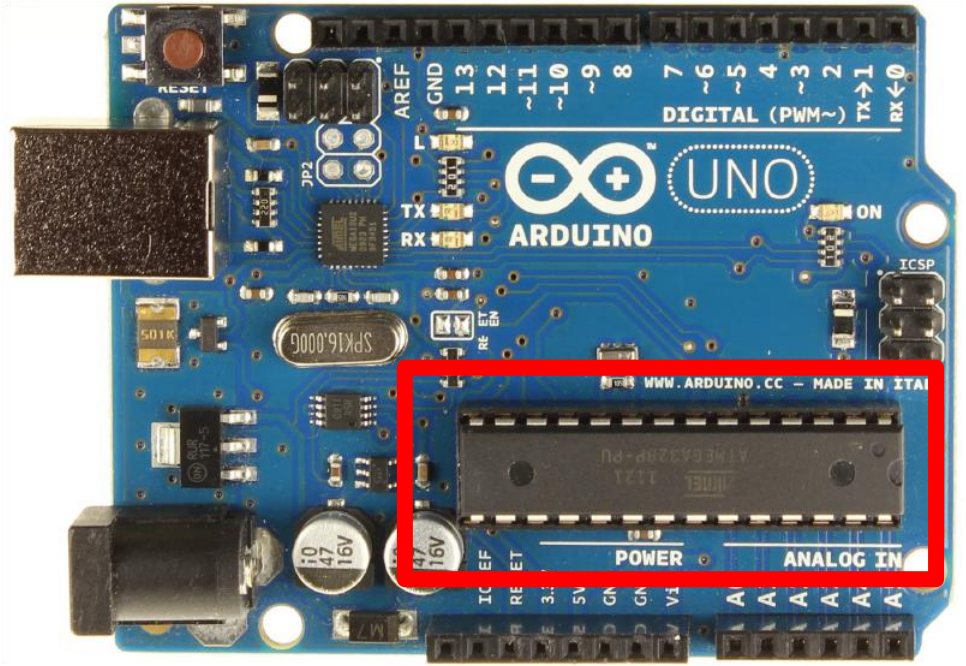
Open source

Large community to support it

# What’s on the board?

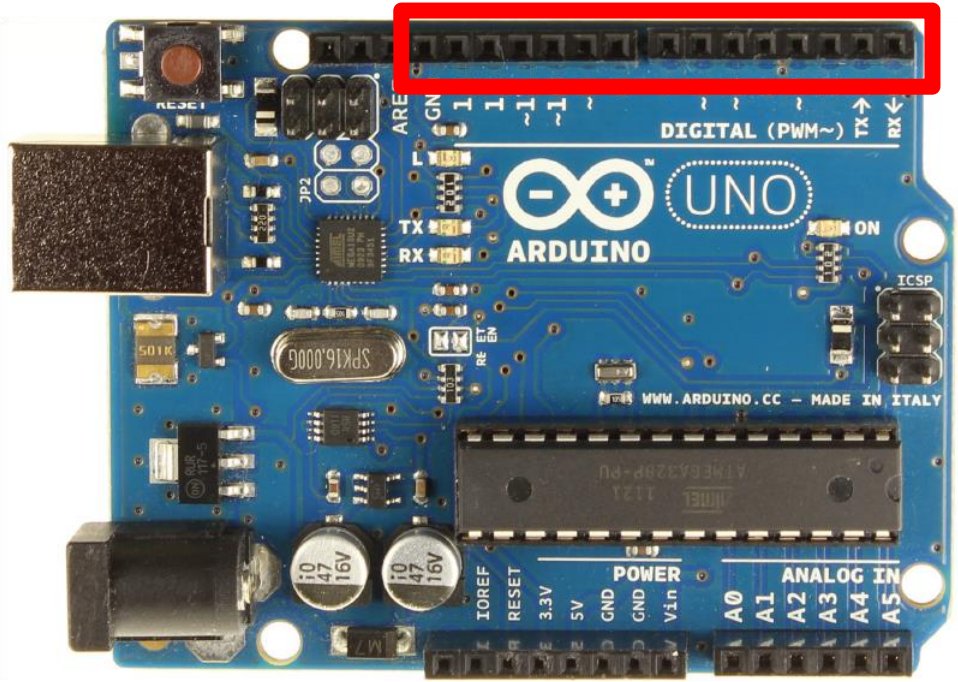


# The Brain - ATmega 328p chip




You can also pull it out.
Careful! Don’t bend the pins

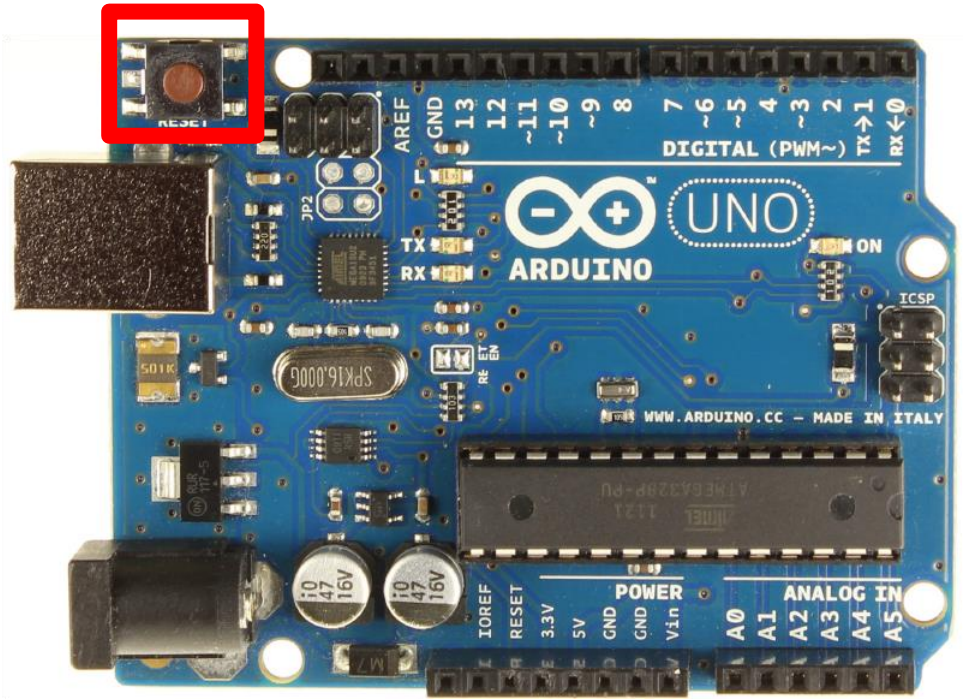
# Digital Pins



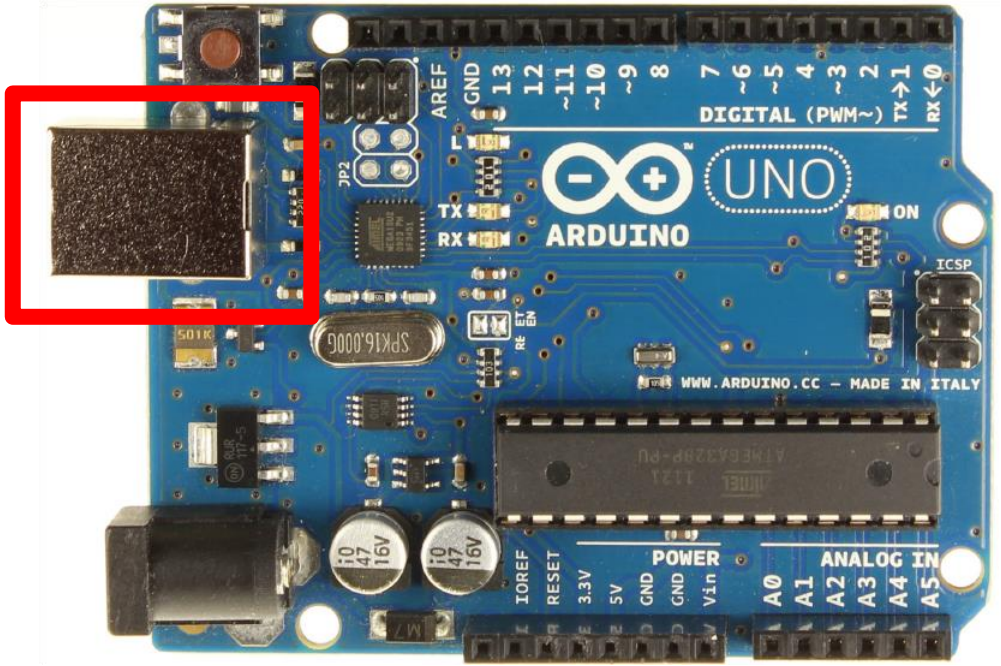
# Analog Pins


DIGITAL (PWM~)
UNO
ARDUINO
ON
ICSP
WWW.ARDUINO.CC – MADE IN ITALY
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
0 1 2 3 4 5

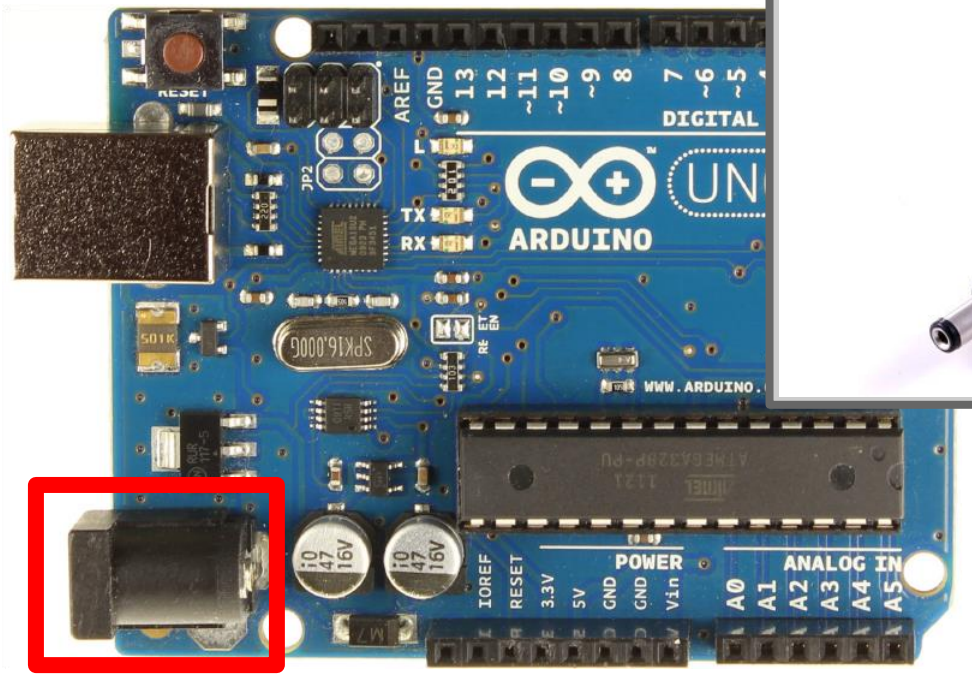
# Reset Button



# USB Port


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

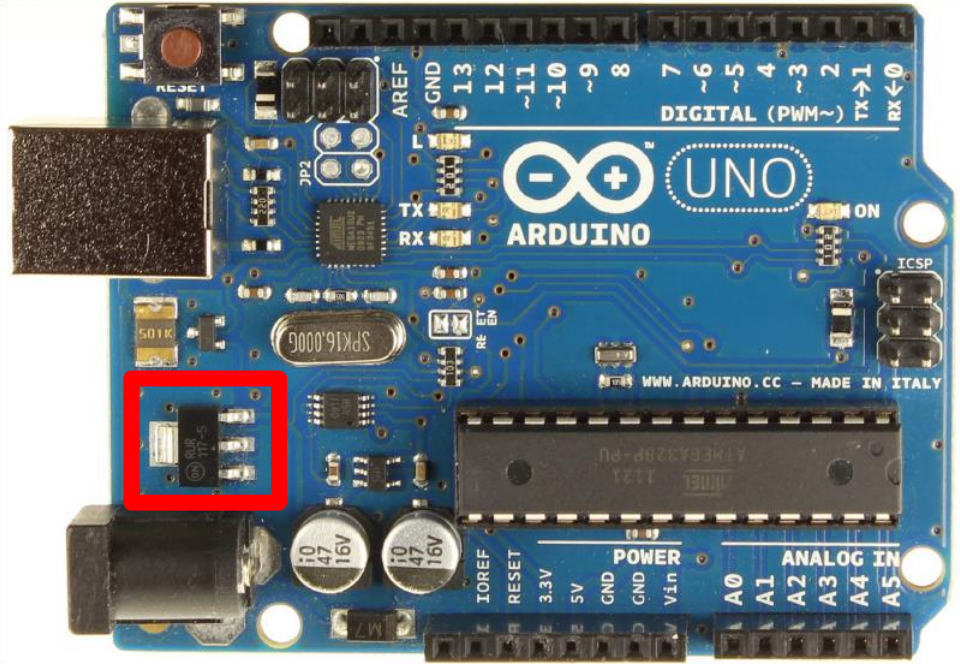
# Power Jack


RESET
AREF
GND
13
12
~11
~10
~9
8
7
~6
~5
DIGITAL
JP2
TX
RX
ARDUINO
UN
WWW.ARDUINO.
POWER
ANALOG IN
IOREF
RESET
3.3V
5V
GND
GND
Vin
A0
A1
A2
A3
A4
A5

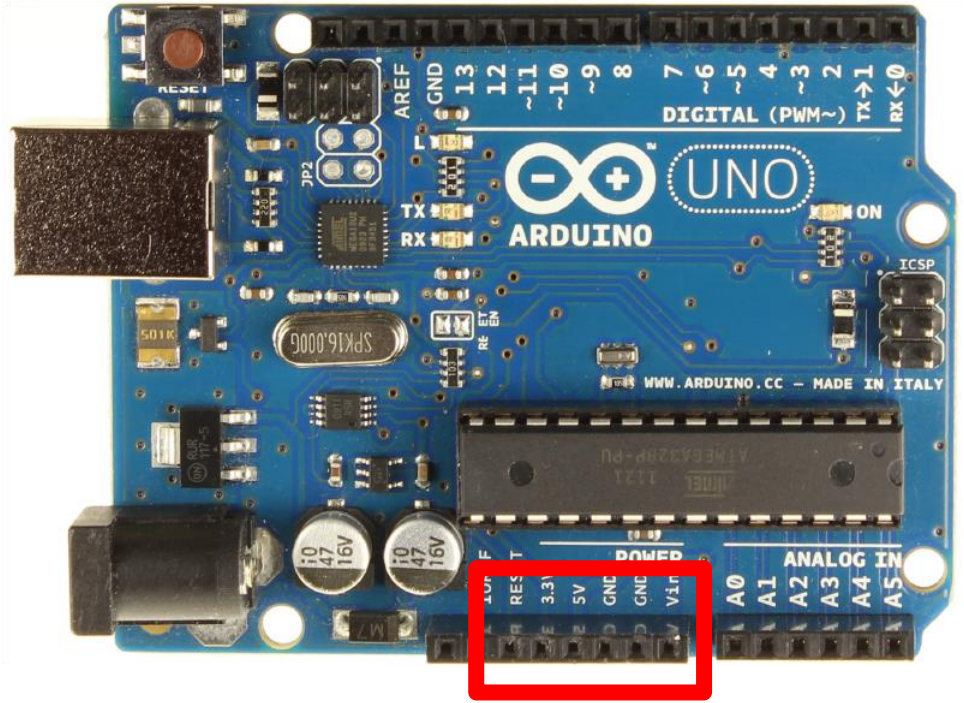
# PRO-TIP: Buy one of these


*Specs: http://playground.arduino.cc/Learning/WhatAdapter*

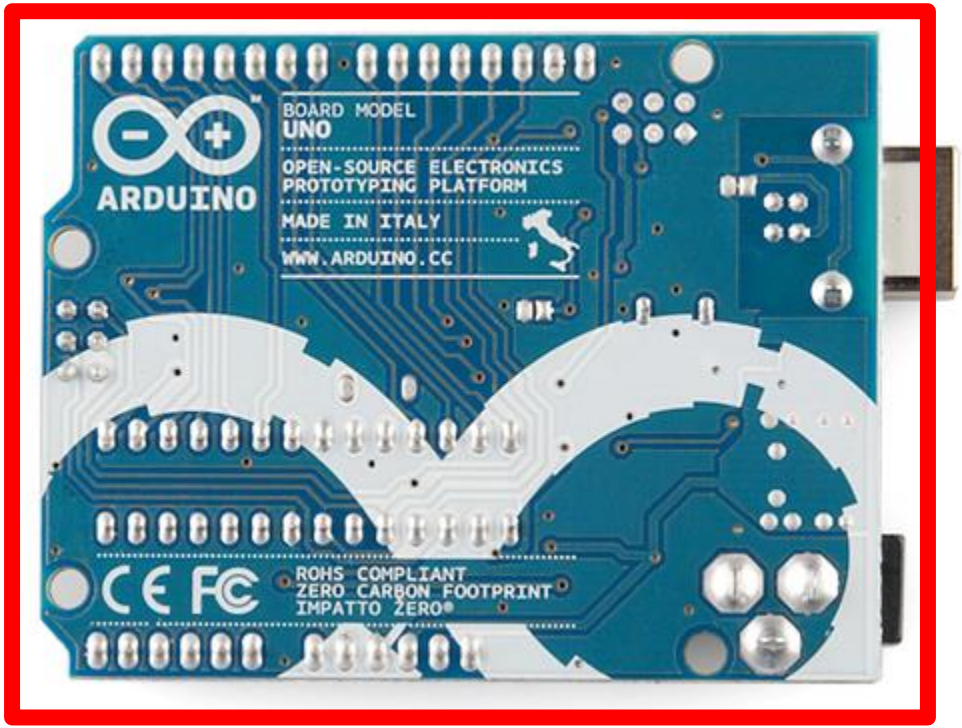
# Voltage Regulator



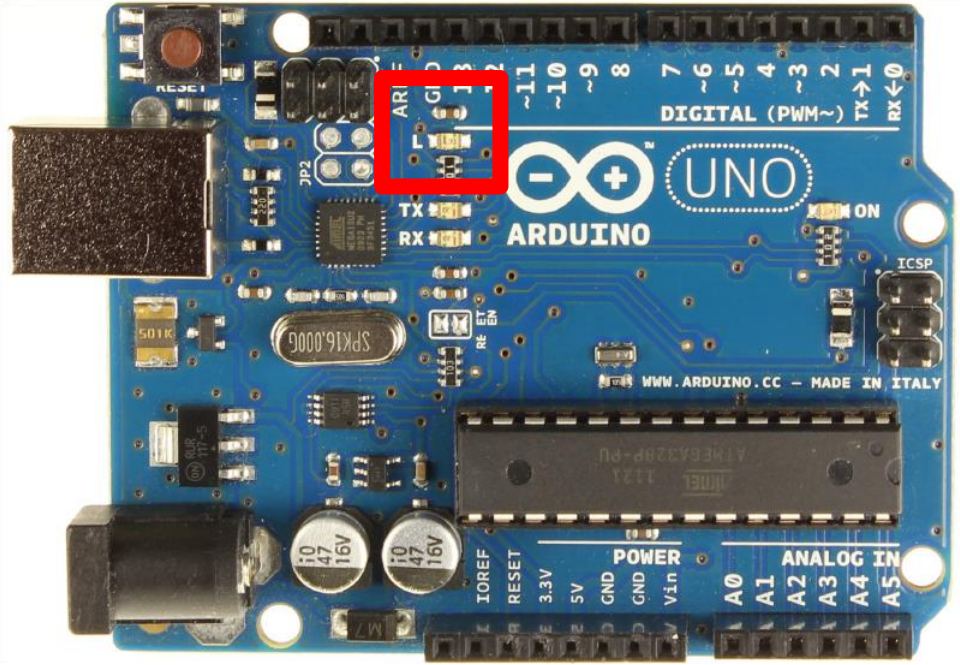
# Vin, 5V, 3.3V, GND, Reset pins



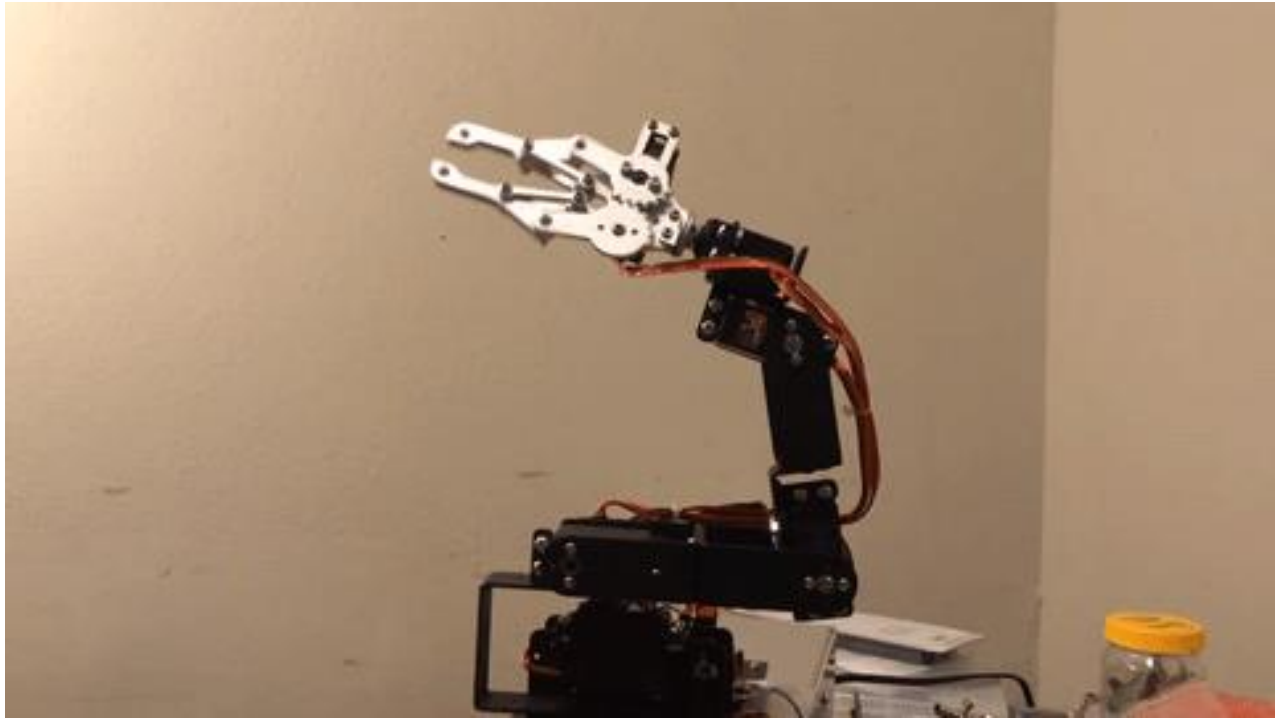
# Back of Arduino



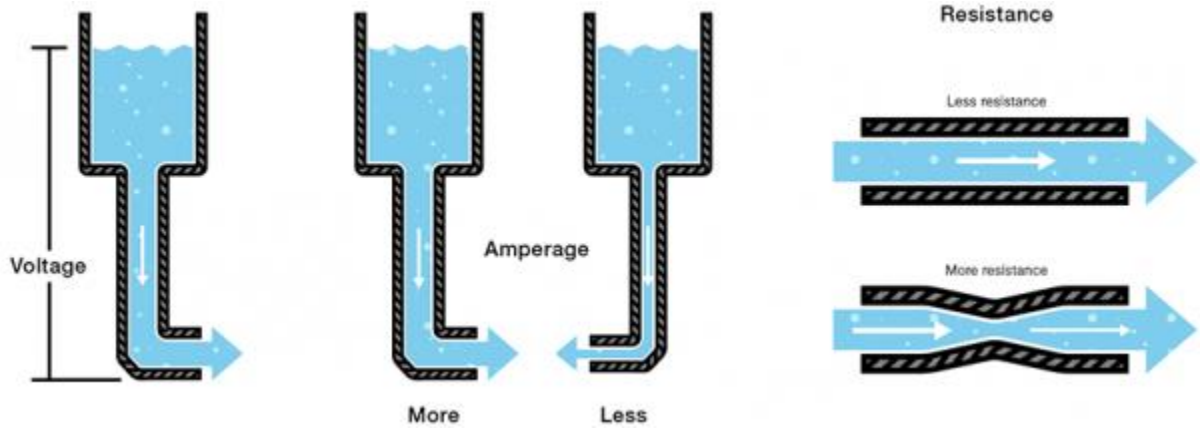
# Internal LED



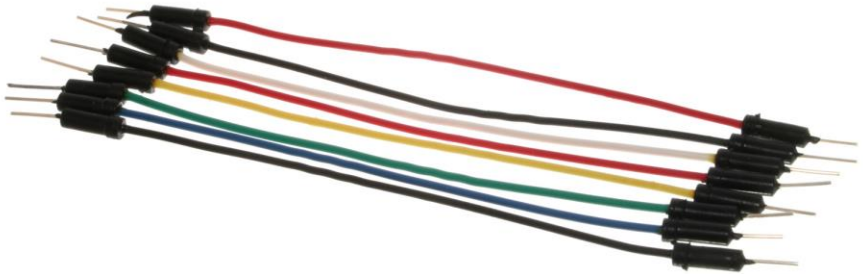
# YAY ELECTRONICS!

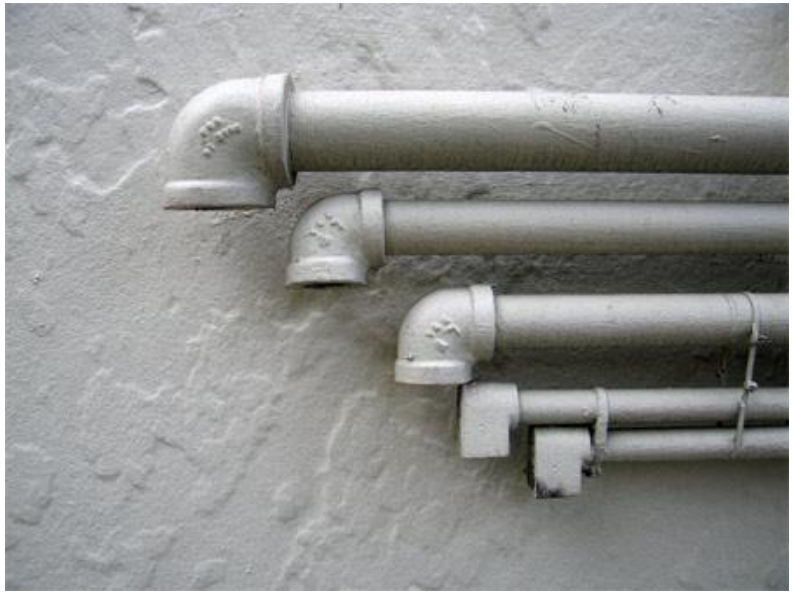


# Ohms Law:
# V = IR


Voltage
Amperage
More
Less
Resistance
Less resistance
More resistance

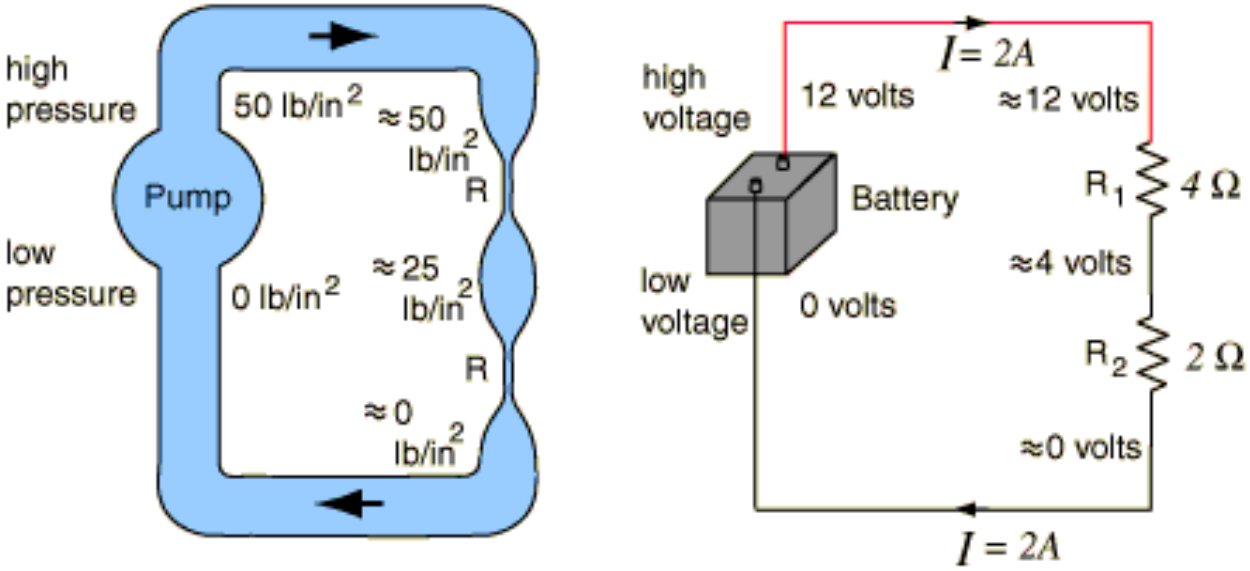
# Wires = Pipes





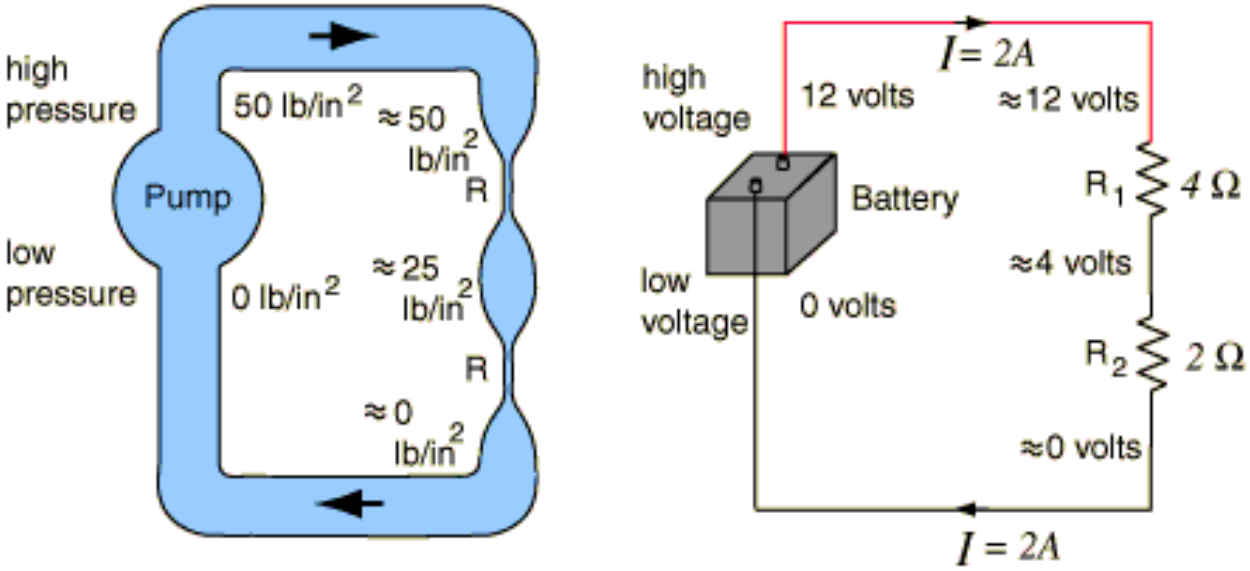
# Battery = Water Pump


# Voltage (V) = water pressure



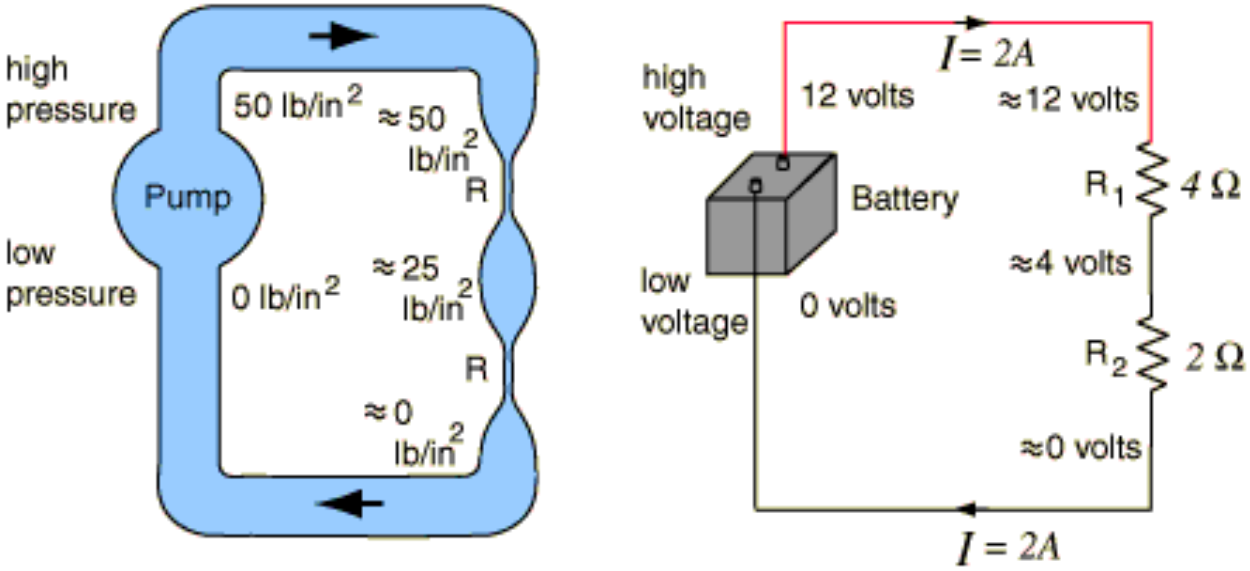

The force with which electrons are being pushed through the wire

# Current (I) = how much water flow



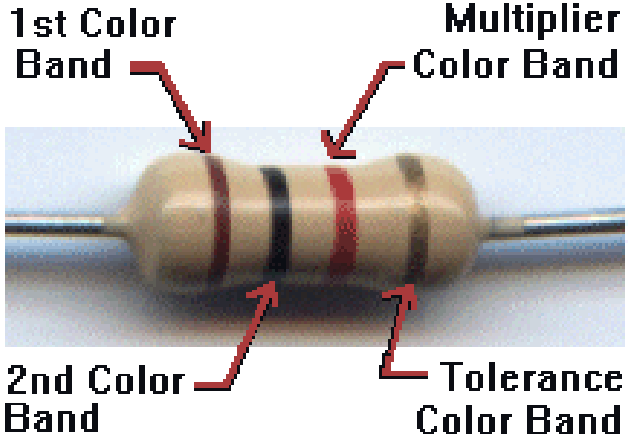

The amount of electrons moving through the wire at any given moment

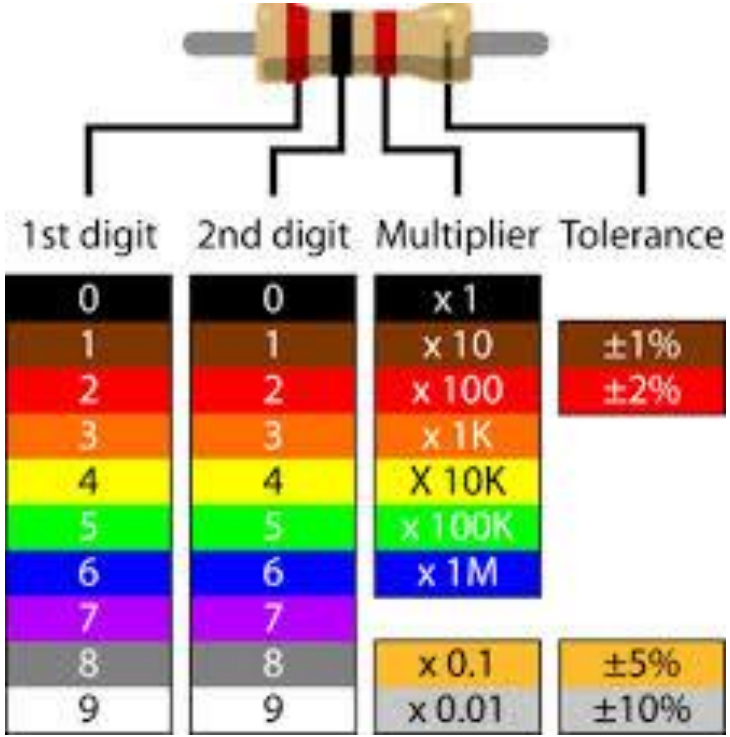
# Resistance (Ohm or Ω) = Pinched pipe



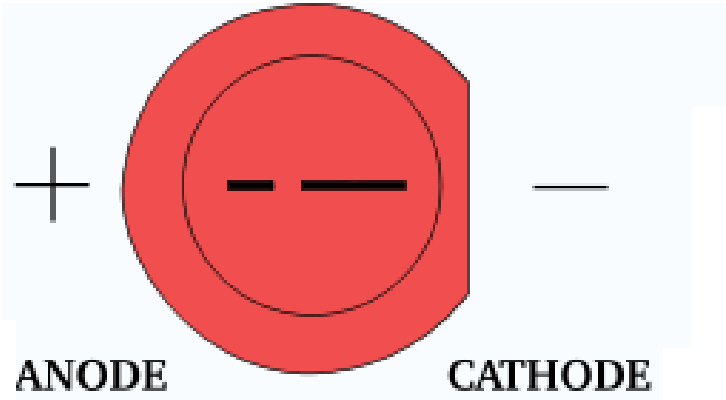

The amount of electrons moving through the wire at any given moment

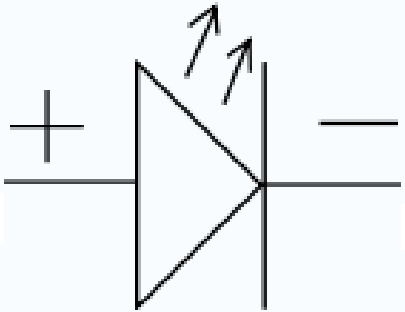
# Resistors

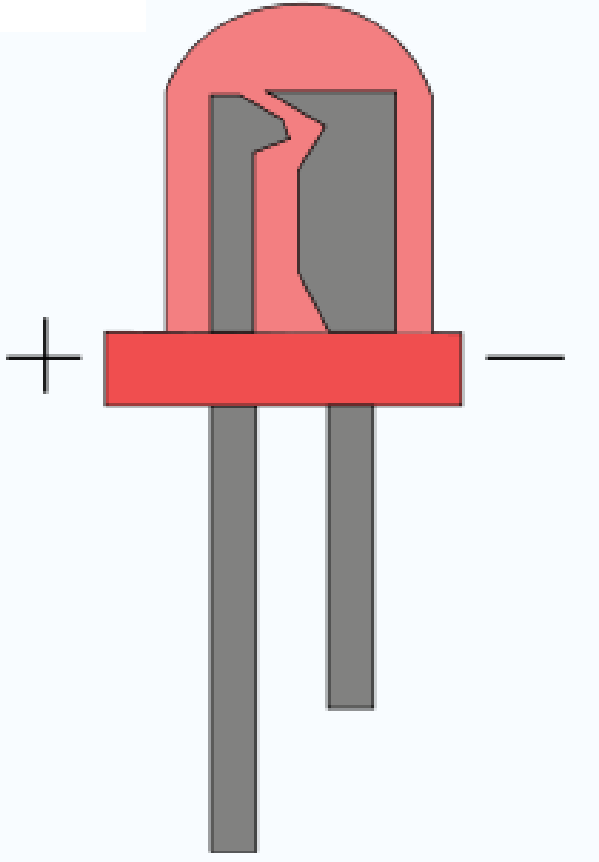

1st Color Band
Multiplier Color Band
2nd Color Band
Tolerance Color Band


1st digit
2nd digit
Multiplier
Tolerance
0
1
2
3
4
5
6
7
8
9
x 1
x 10
x 100
x 1K
X 10K
x 100K
x 1M
x 0.1
x 0.01
±1%
±2%
±5%
±10%

# LEDs

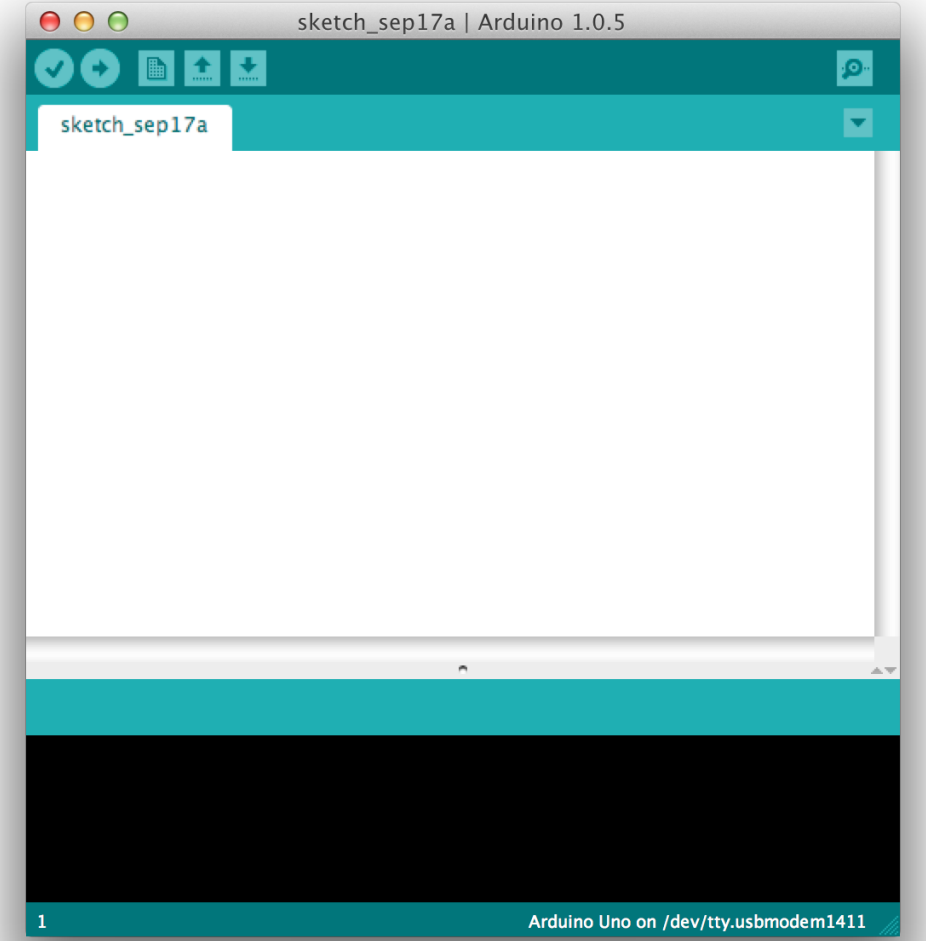




# Arduino IDE

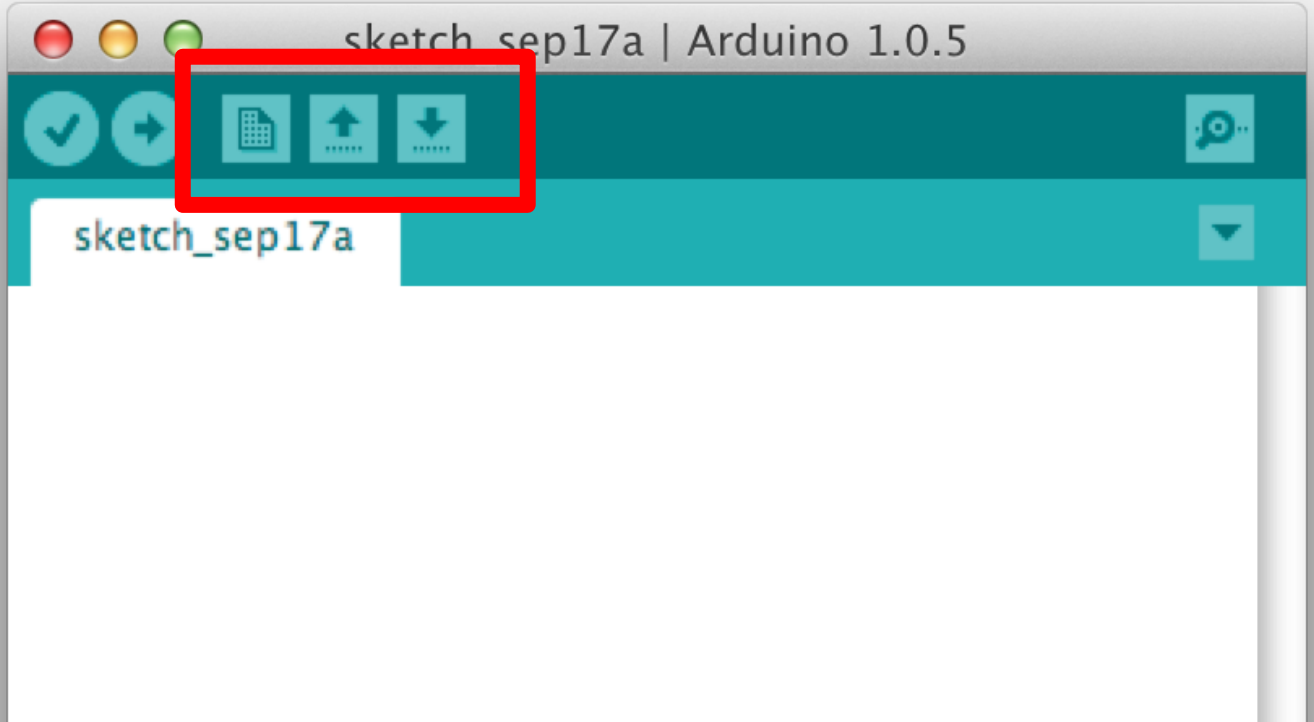
**(Integrated Dev Environment)**



# Verify Button

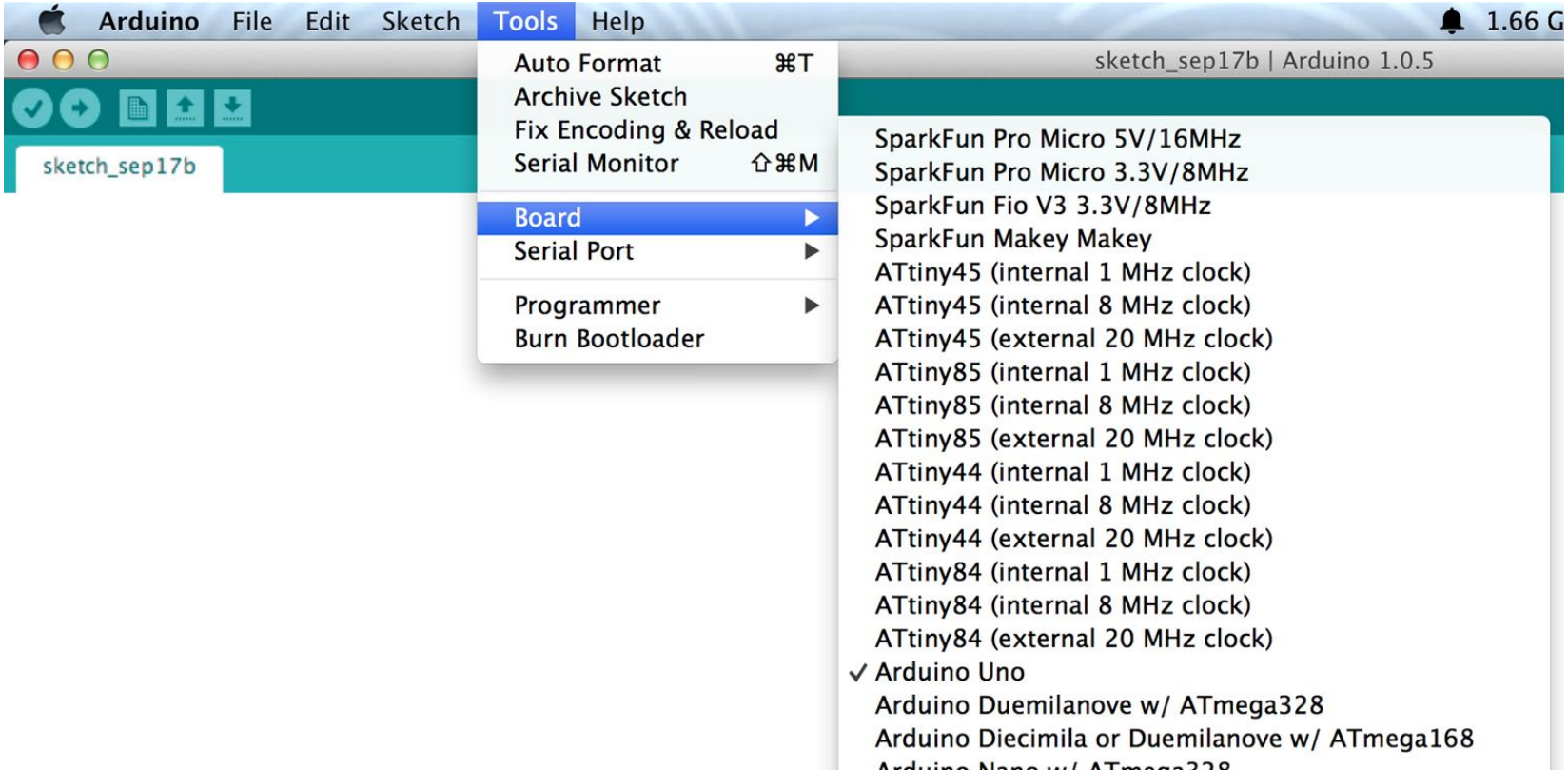
# Upload Button

# New, Open, Save...

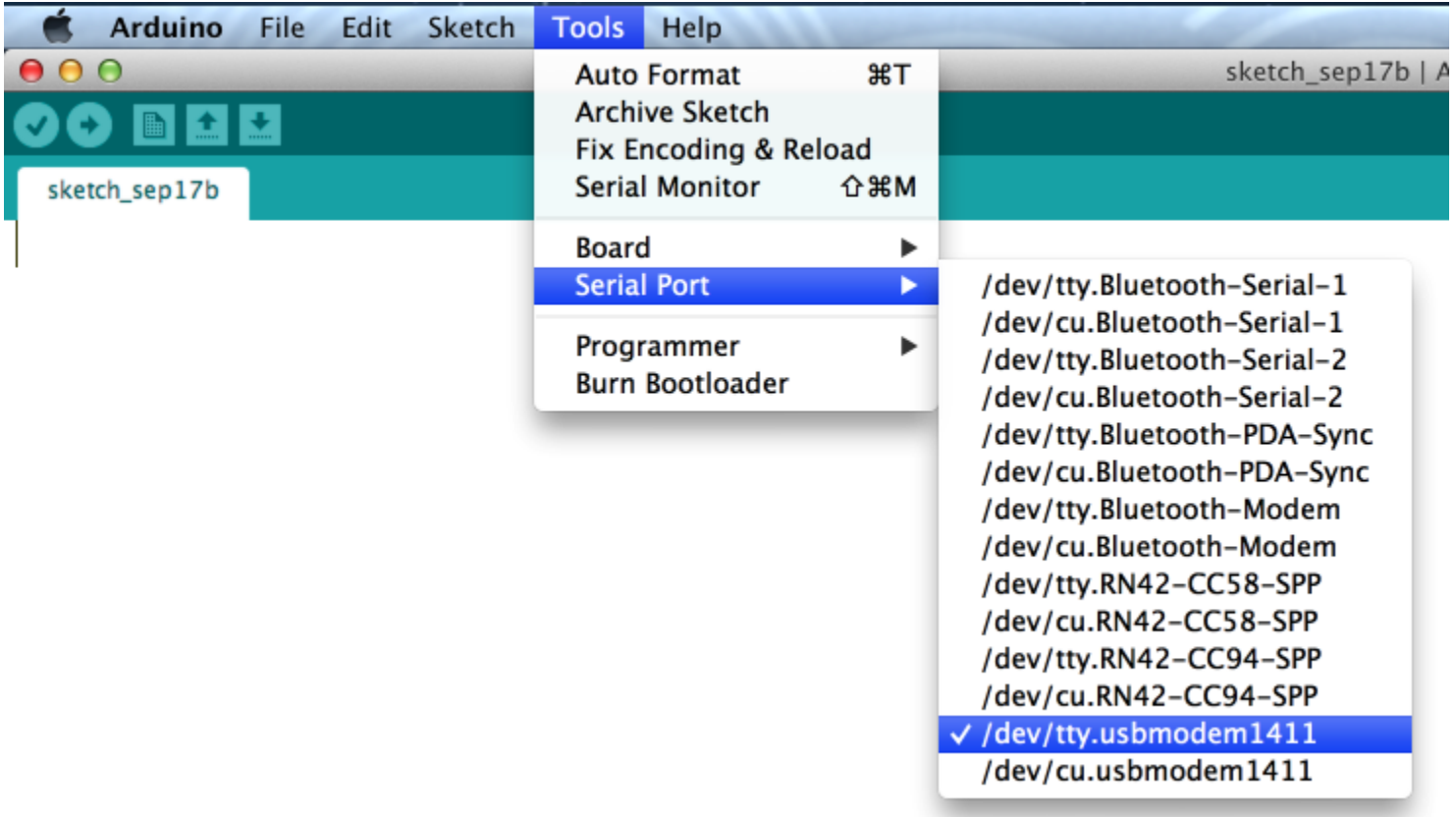


# Serial Monitor

# Tools > Board


Arduino
File
Edit
Sketch
Tools
Help
1.66 G
sketch_sep17b | Arduino 1.0.5
sketch_sep17b
Auto Format ⌘T
Archive Sketch
Fix Encoding & Reload
Serial Monitor ⇧⌘M
Board
Serial Port
Programmer
Burn Bootloader
SparkFun Pro Micro 5V/16MHz
SparkFun Pro Micro 3.3V/8MHz
SparkFun Fio V3 3.3V/8MHz
SparkFun Makey Makey
ATtiny45 (internal 1 MHz clock)
ATtiny45 (internal 8 MHz clock)
ATtiny45 (external 20 MHz clock)
ATtiny85 (internal 1 MHz clock)
ATtiny85 (internal 8 MHz clock)
ATtiny85 (external 20 MHz clock)
ATtiny44 (internal 1 MHz clock)
ATtiny44 (internal 8 MHz clock)
ATtiny44 (external 20 MHz clock)
ATtiny84 (internal 1 MHz clock)
ATtiny84 (internal 8 MHz clock)
ATtiny84 (external 20 MHz clock)
✓ Arduino Uno
Arduino Duemilanove w/ ATmega328
Arduino Diecimila or Duemilanove w/ ATmega168

# Tools > Serial Port



# File > Examples > Basics > Blink sketch

```
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

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

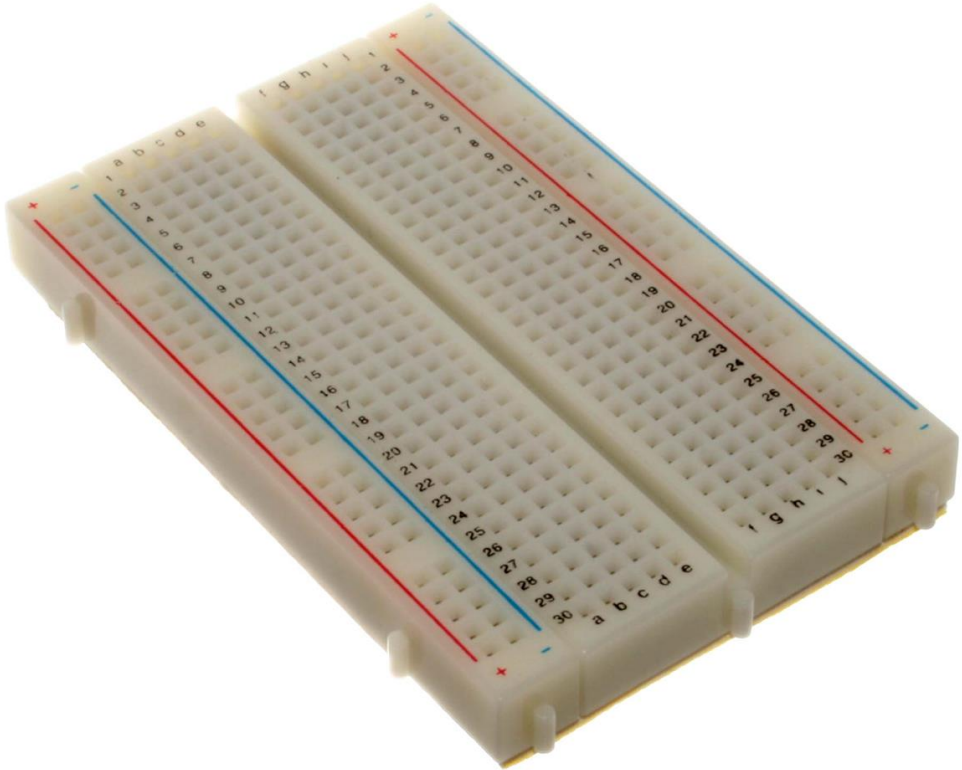
# File > Examples > Basics > Blink sketch

```
// Pin 13 has an LED connected on most Arduino boards.
// give it a name:
int led = 13;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

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

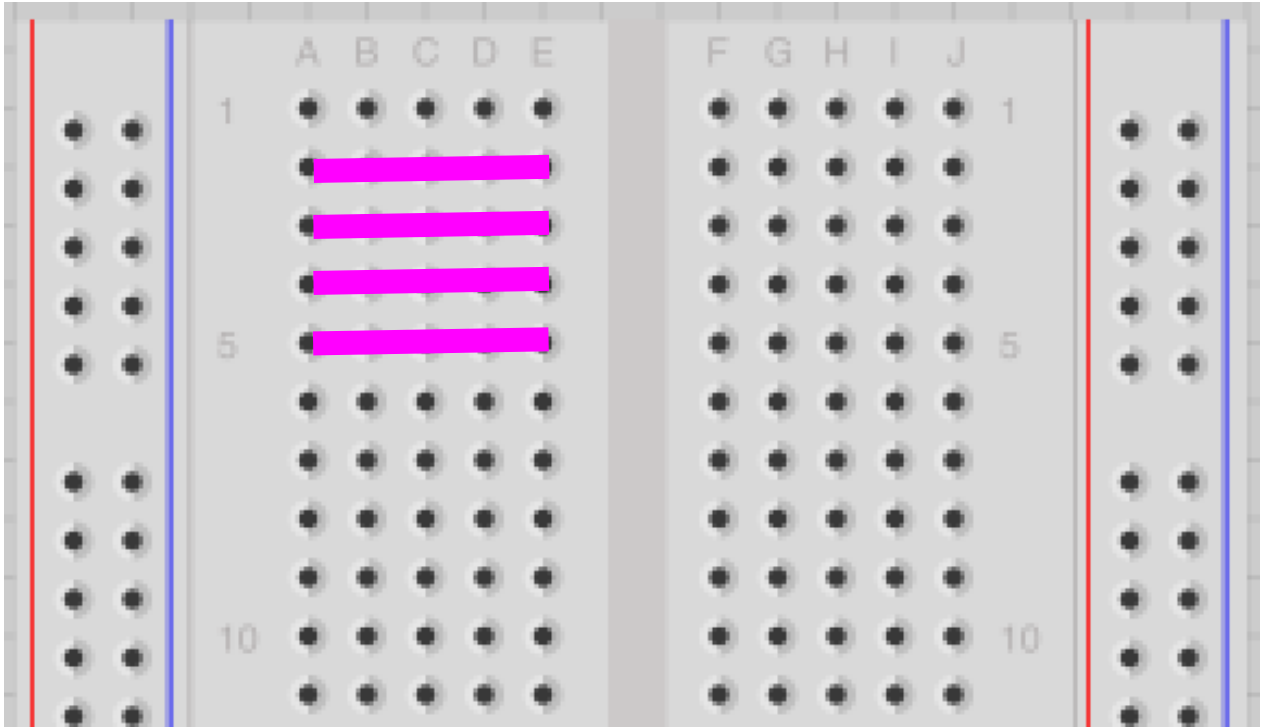
# Add some “debugging” code

```
void setup() {
  //start the serial connection from Arduino back to computer
  Serial.begin(9600);

  // initialize the digital pin as an output.
  pinMode(led, OUTPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
  Serial.println("LED is On");
  delay(1000);               // wait for a second
  digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
  Serial.println("LED is Off");
  delay(1000);               // wait for a second
}
```

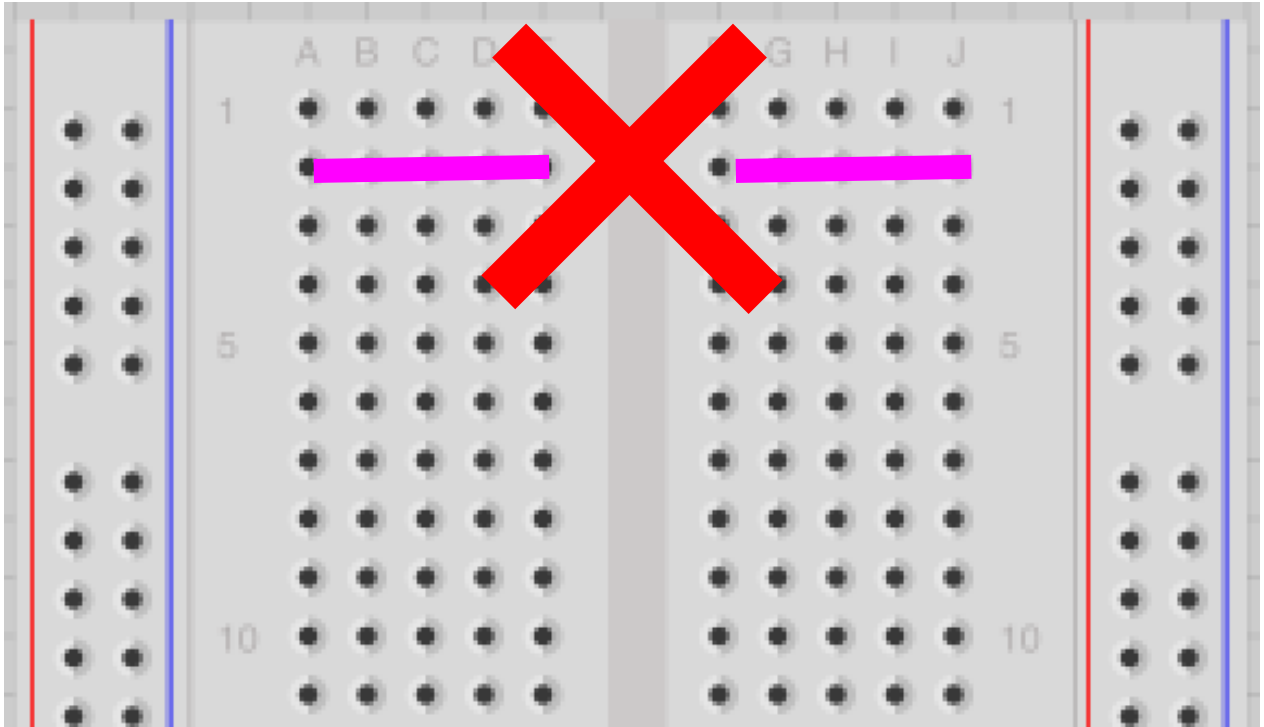
# Breadboarding



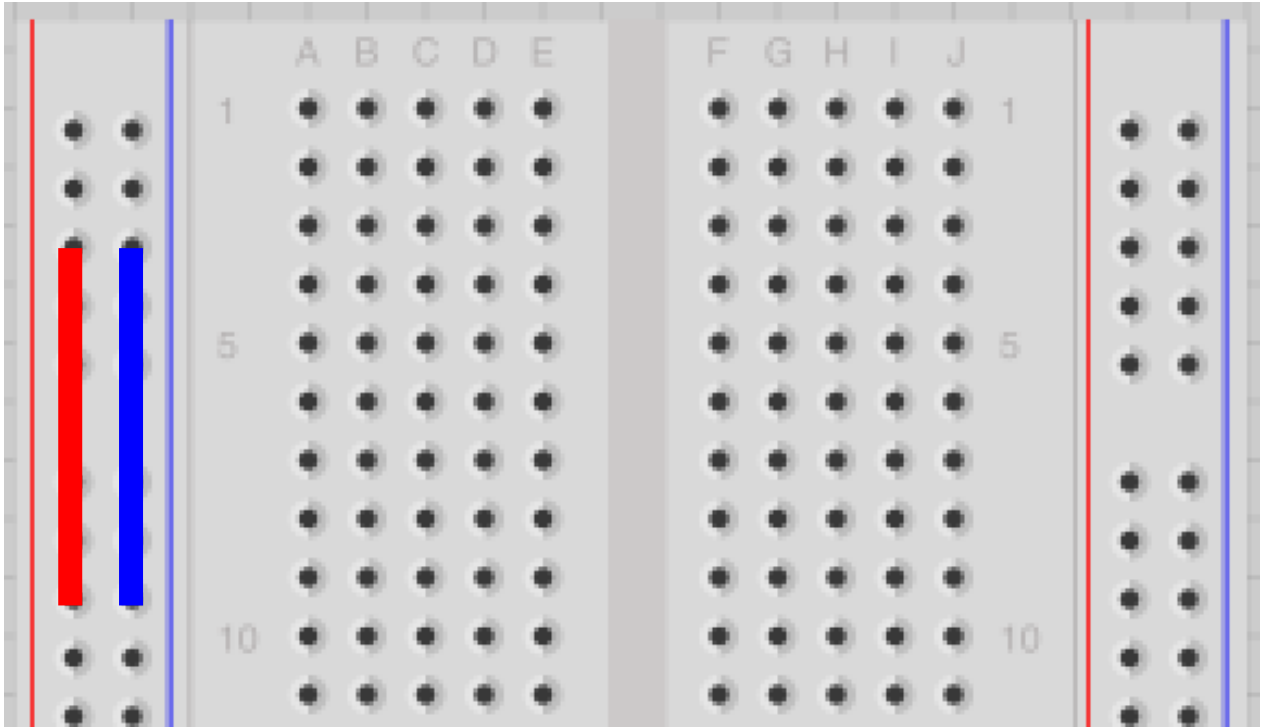
# Breadboard Connections



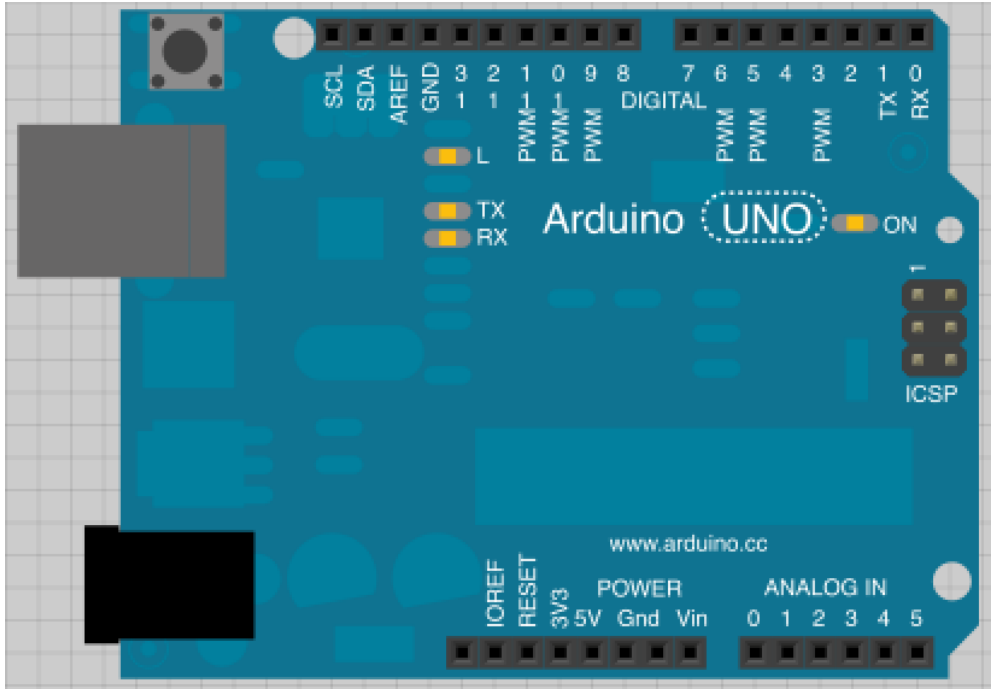
# Breadboard Rows



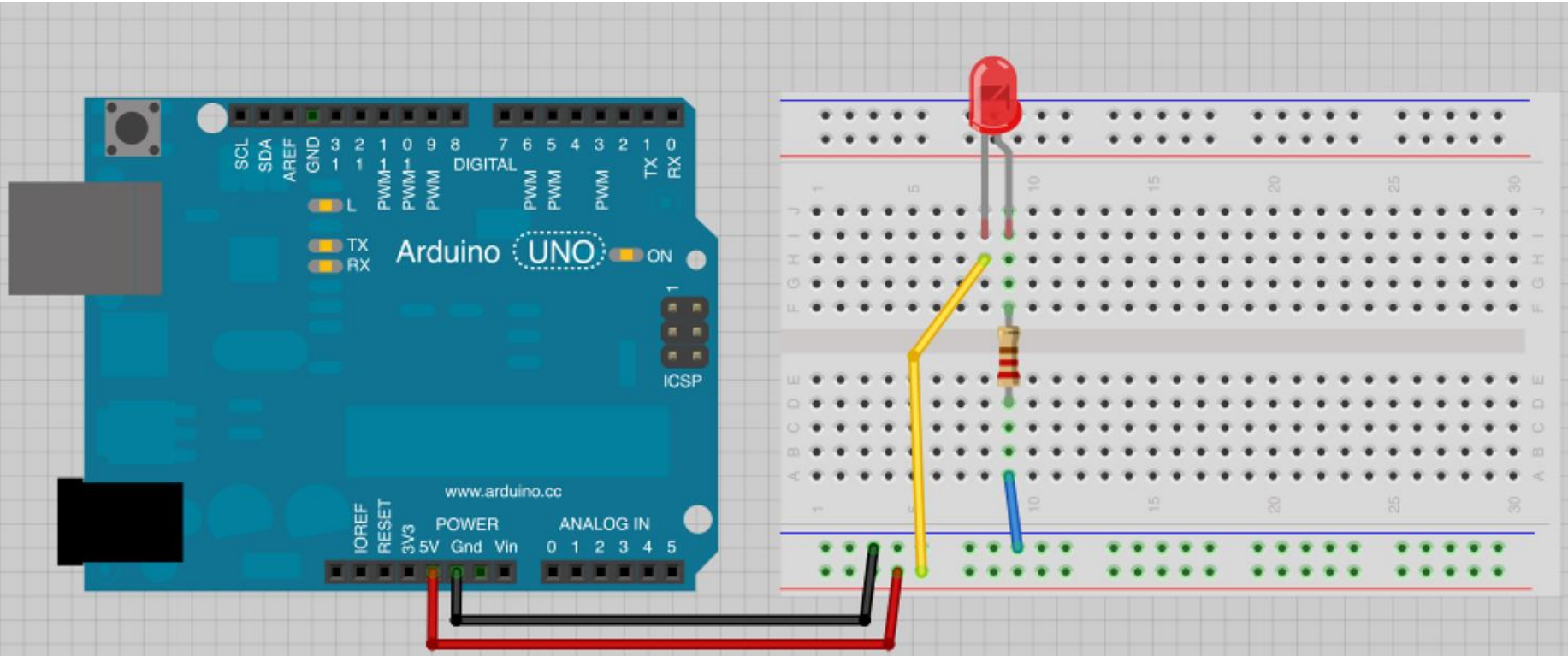
# Breadboard Power/Ground “Rails”



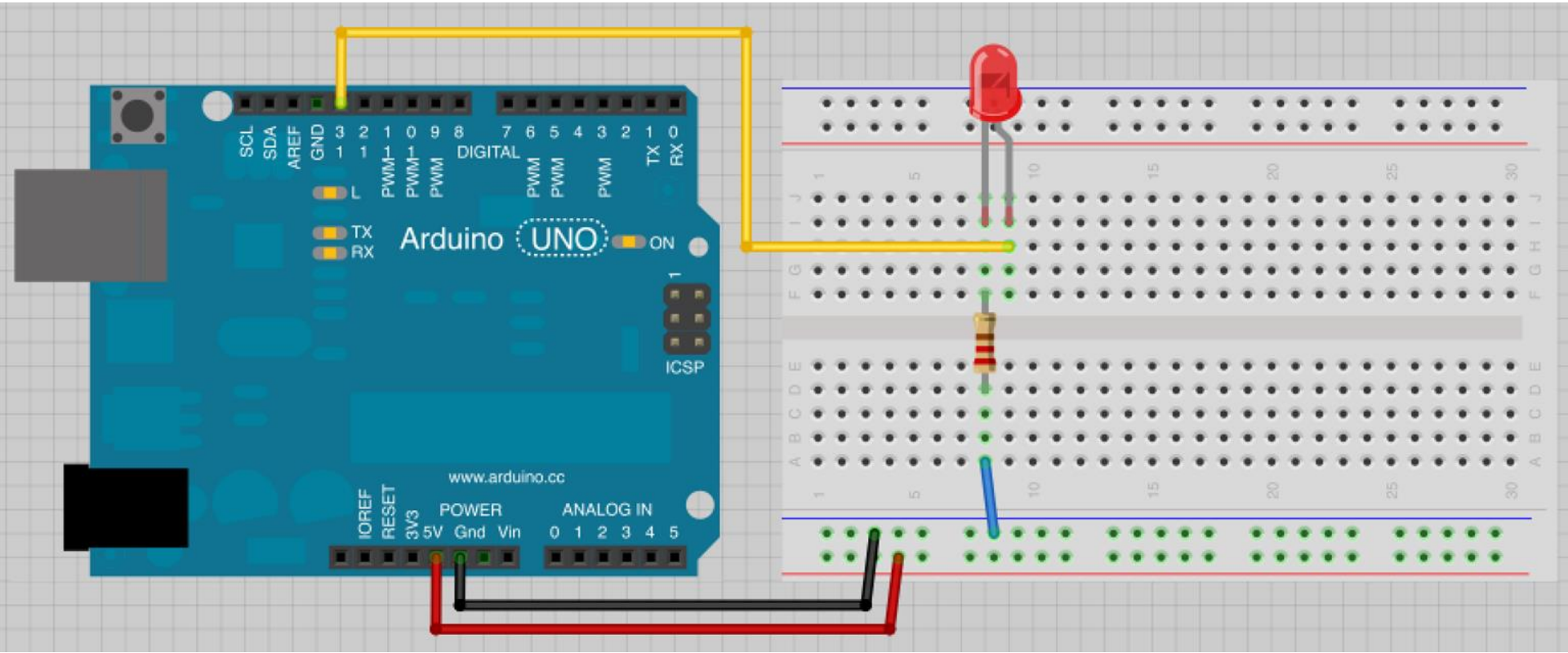
# Connecting an LED



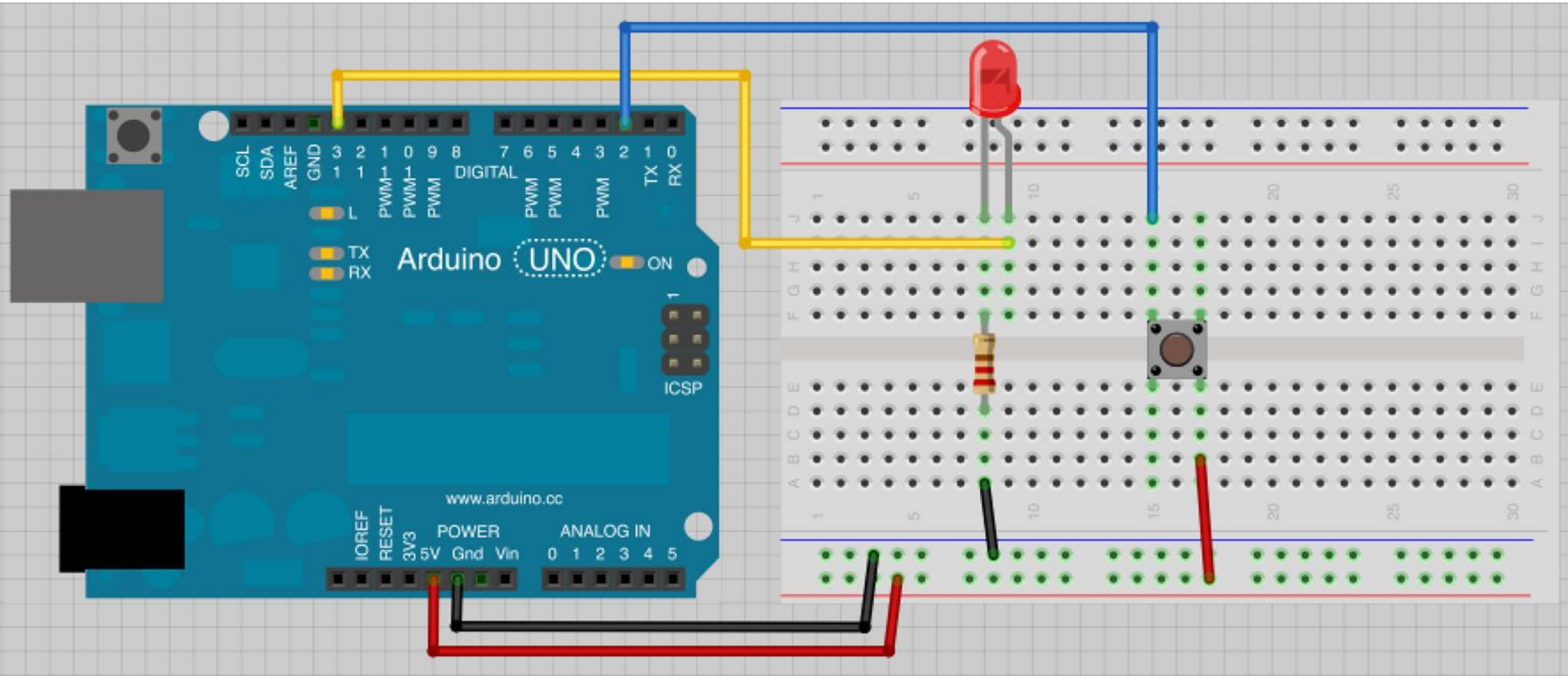
# Connecting an LED



# Control blink via Pin 13



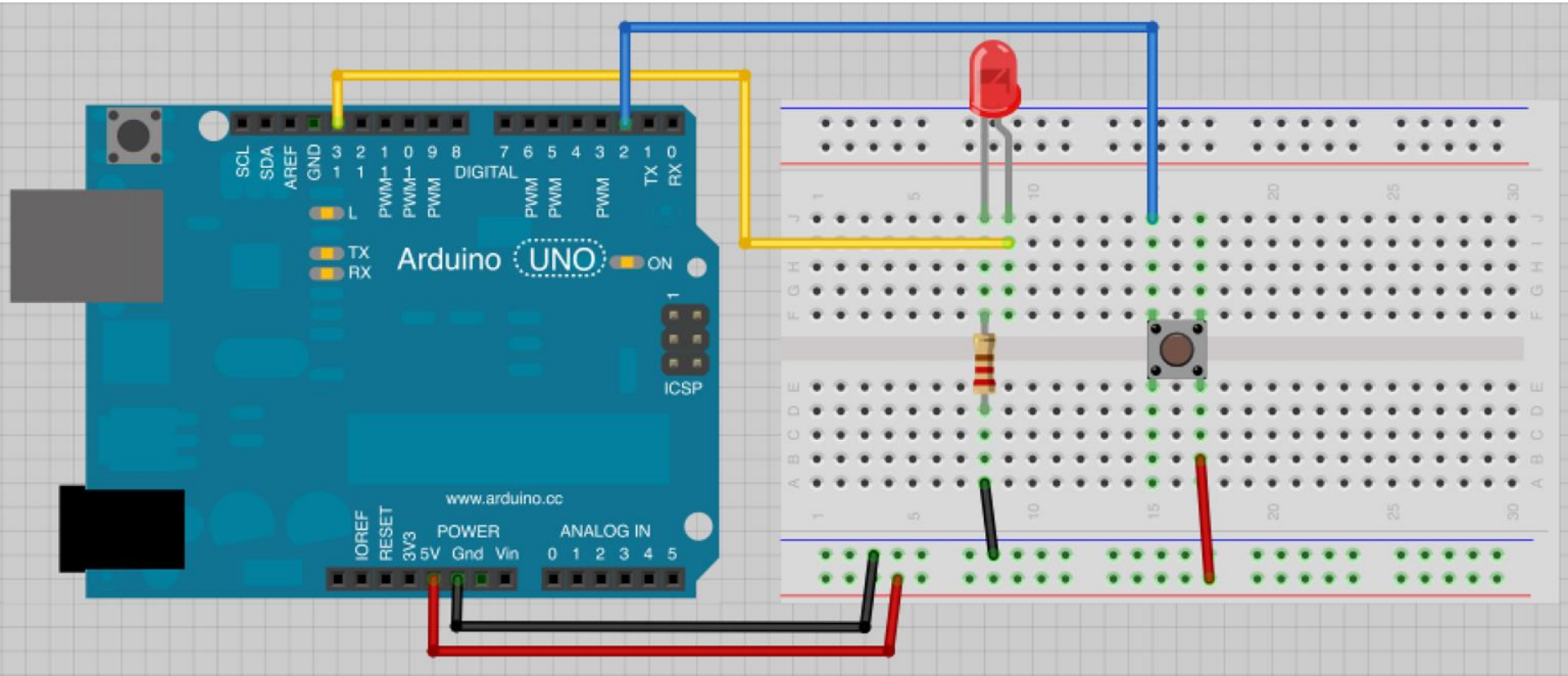
# Manually blink using push button.



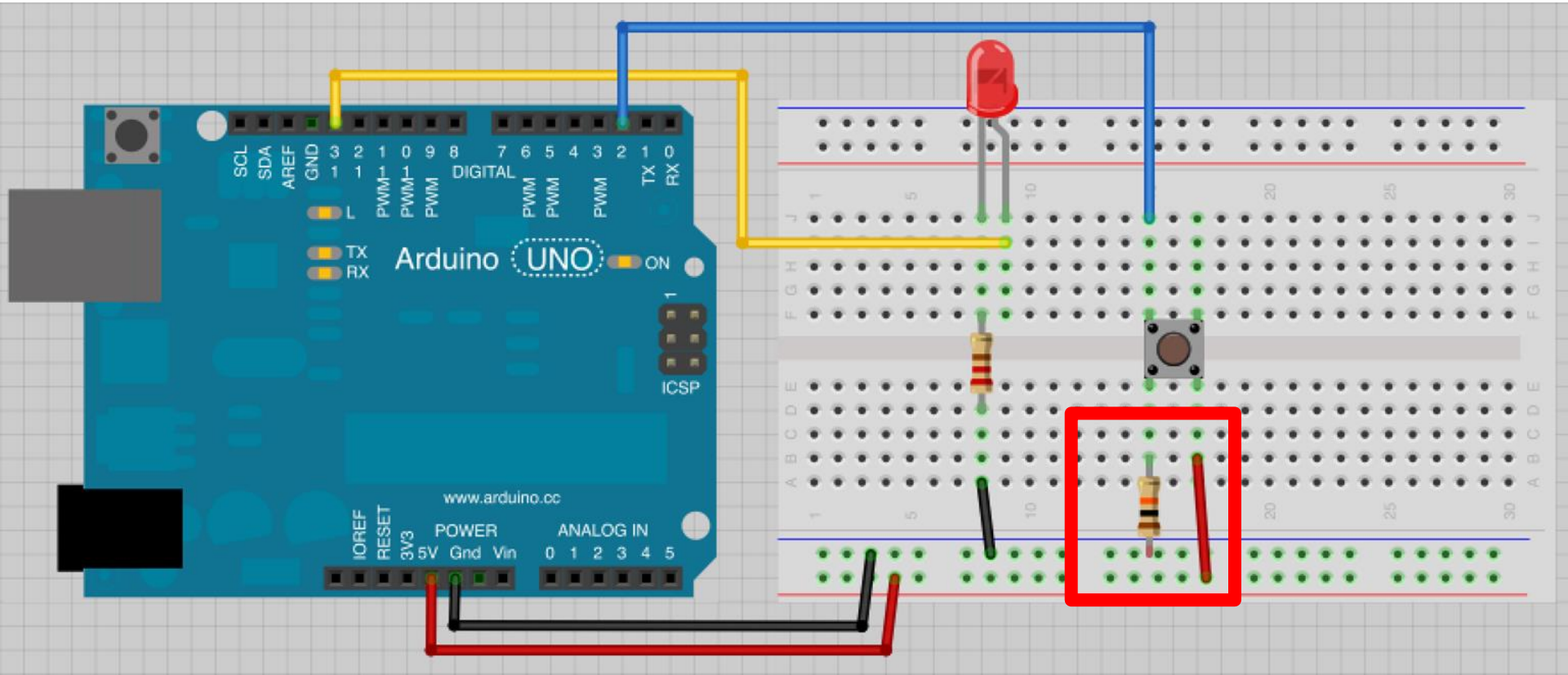
# Read the button with code

```
int led = 13;
int buttonPin = 2;
int buttonState = 0;

// the setup routine runs once when you press reset:
void setup() {
  // initialize the pinModes
  pinMode(led, OUTPUT);
  pinMode(buttonPin, INPUT);
}

// the loop routine runs over and over again forever:
void loop() {
  //read the button
  buttonState = digitalRead(buttonPin);
  //Perform different actions depending on the state of the button
  if(buttonState == HIGH){
    digitalWrite(led, HIGH);   // turn the LED on (HIGH is the voltage level)
    delay(1000);               // wait for a second
  } else {
    digitalWrite(led, LOW);    // turn the LED off by making the voltage LOW
    delay(1000);               // wait for a second
  }
}
```

# Connecting a Button

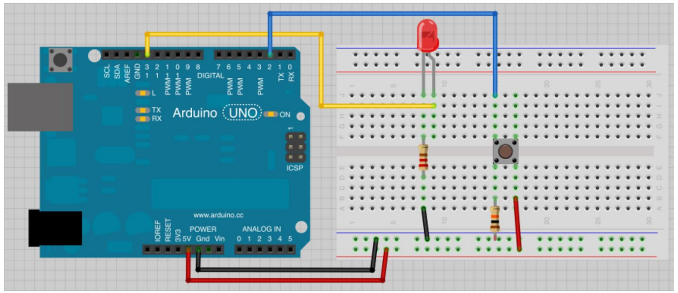


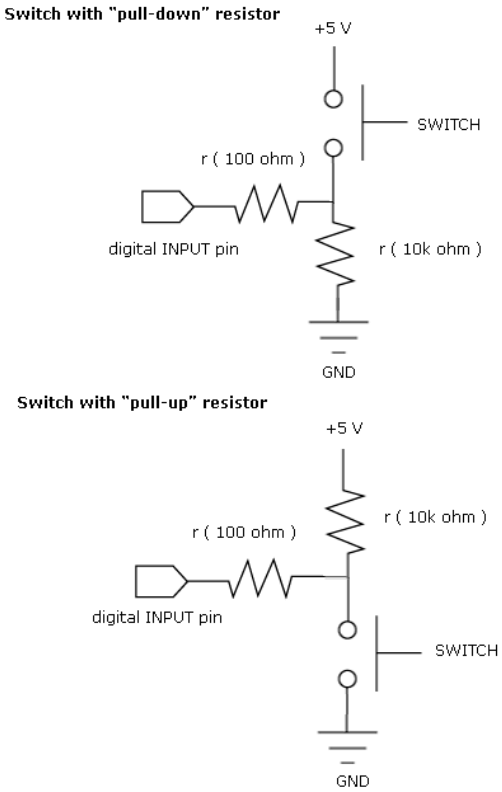
# Add a “pull-down” resistor



# Pull-Up / Pull-Down Resistors

ensure that the signal will be a valid logic level if external devices are disconnected

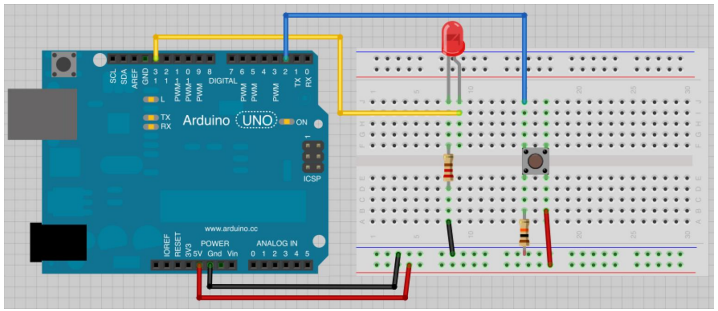


# HOMEWORK-github folder structure

**~ -> week1 -> index.html**

# HOMEWORK




**Update your code so the button triggers a state change**

For example: the LED stays on when you push it and turns off when the button is pressed again

Try out different blink patterns

**Documentation:**
Upload your code to github with a link to the online video