

# Embedded System Workshops

---

01. Blinking LEDs  
*CCA Girls Who Code*



# Project Overview

- Purpose:
  - ◆ Introduce circuits, proper circuit design, and circuit elements
  - ◆ Learn the Arduino IDE
  - ◆ Learn to send digital outputs
- Projects
  - ◆ Single LED  $\Rightarrow$  Multiple LED circuit
- Grab your kit, and let's get started!

# Parts List

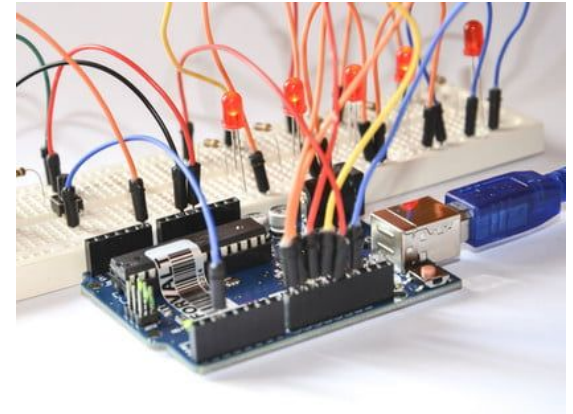
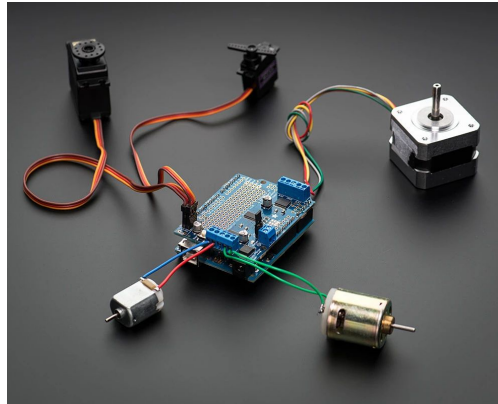
Below is the list of parts we'll be using during this lesson

- Arduino UNO R3 Controller Board
- USB Cable
- Breadboard
- LEDs (all colors!)
- 220 $\Omega$  Resistors
- Male-male jumper wires

# What is Arduino?



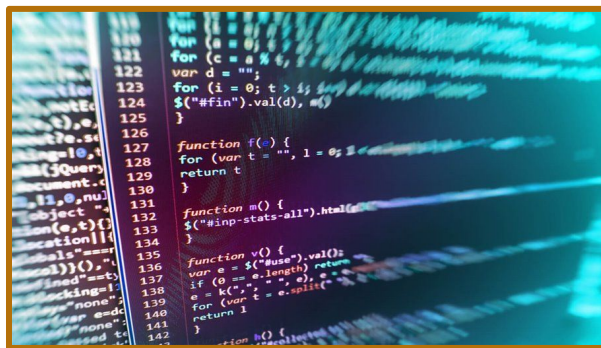
- A small microcontroller used for a variety of applications
- Can be programmed using C and C++ programming languages
- Open source; many Arduino-compatible and Arduino-derived boards exist



# What is C++?

C++ is a popular programming language used for a variety of things, such as

- Embedded systems
- Developing system and desktop applications
- Developing operating systems such as Apple's OS X and Microsoft's Windows
- Compiler production
- Open source software



# Basic Circuit Design

A circuit forms a loop that electricity can flow through. Electricity will always flow from high voltage to low voltage.

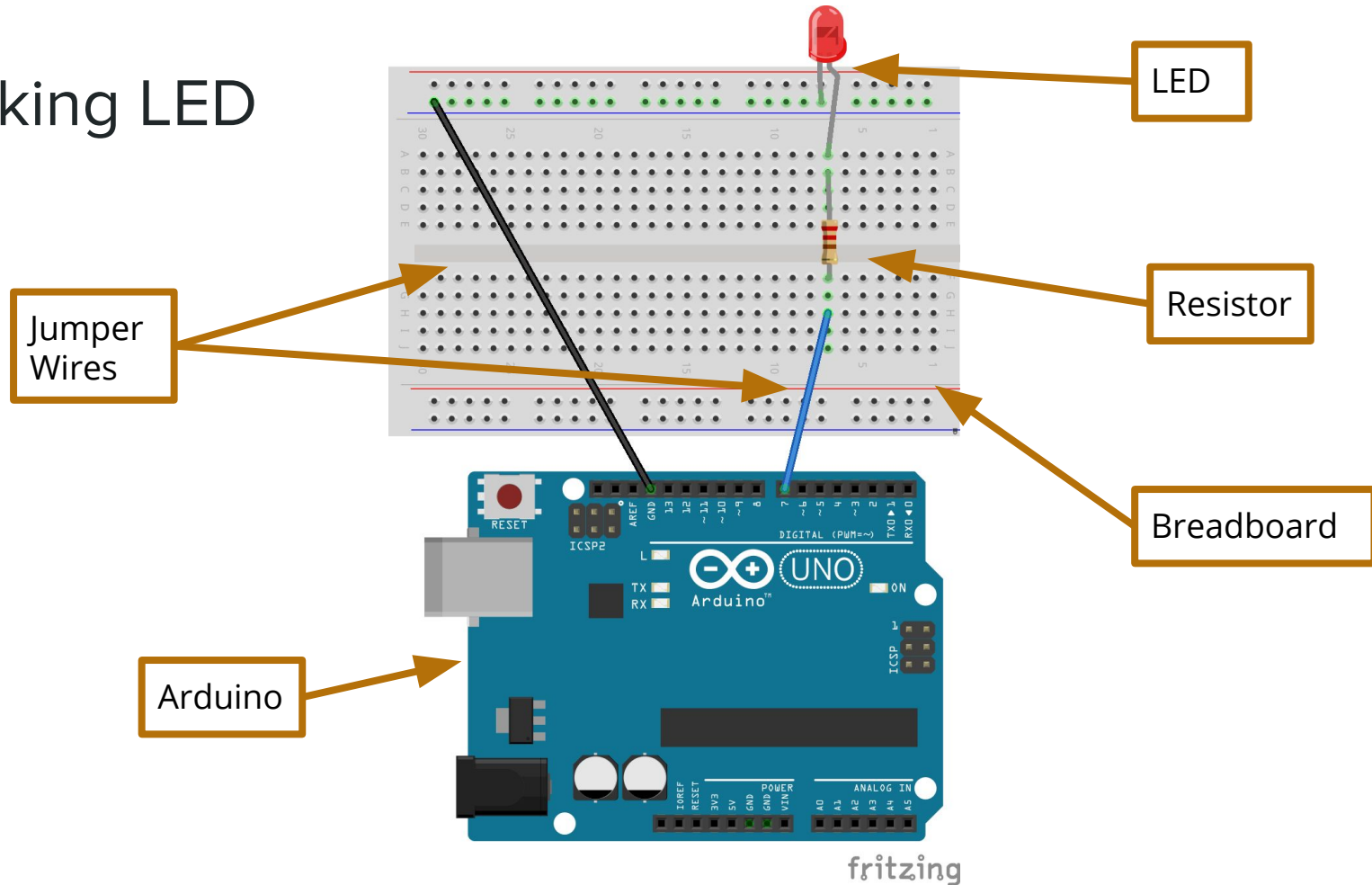
## Circuit Elements:

- Voltage source: This is what provides the power to any circuit. Arduino has two voltage sources: 5 volts and 3.3 volts. Any of the pins can set to 3.3v using the code.
- Ground: The lowest voltage point in a circuit. All circuits should be designed to end at a ground pin.
- Loads: Anything that causes resistance in the circuit, such as actual resistors, LED lights, motors, screens, etc.

# Possible Safety Hazards:

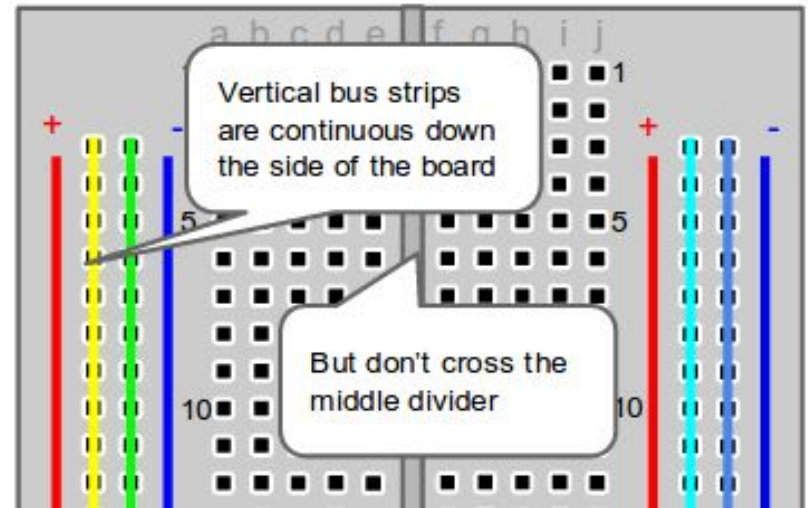
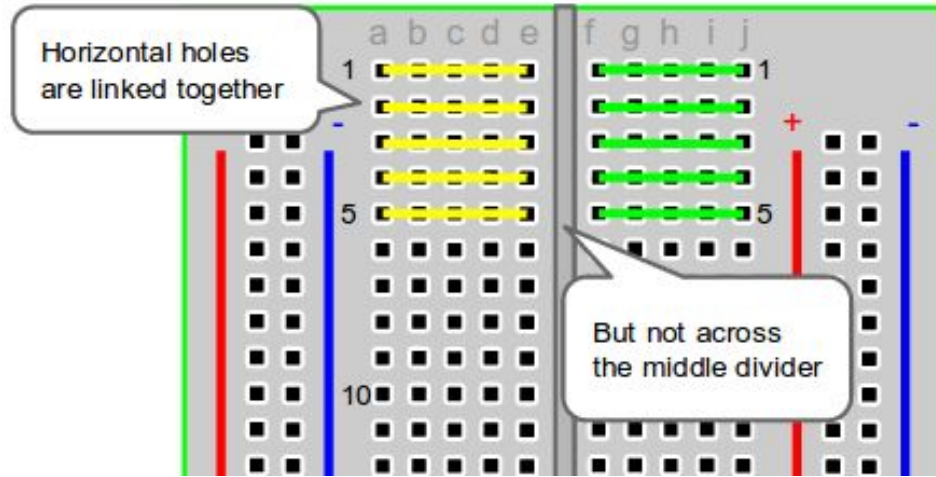
- High Currents:
  - If the current in a circuit is too high due to inadequate or nonexistent resistors, it can damage parts in a circuit, such as LEDs, Motors, or the Arduino itself (this is why circuit breakers exist).
- Short Circuit
  - An electrical short occurs when the electrical current can take a much lower resistance path than what is intended, causing much higher currents than what is intended and potentially damaging parts of the circuit.
  - To prevent this from occurring, make sure all your wires are where you intend for them to go before powering the circuit on.

# Blinking LED



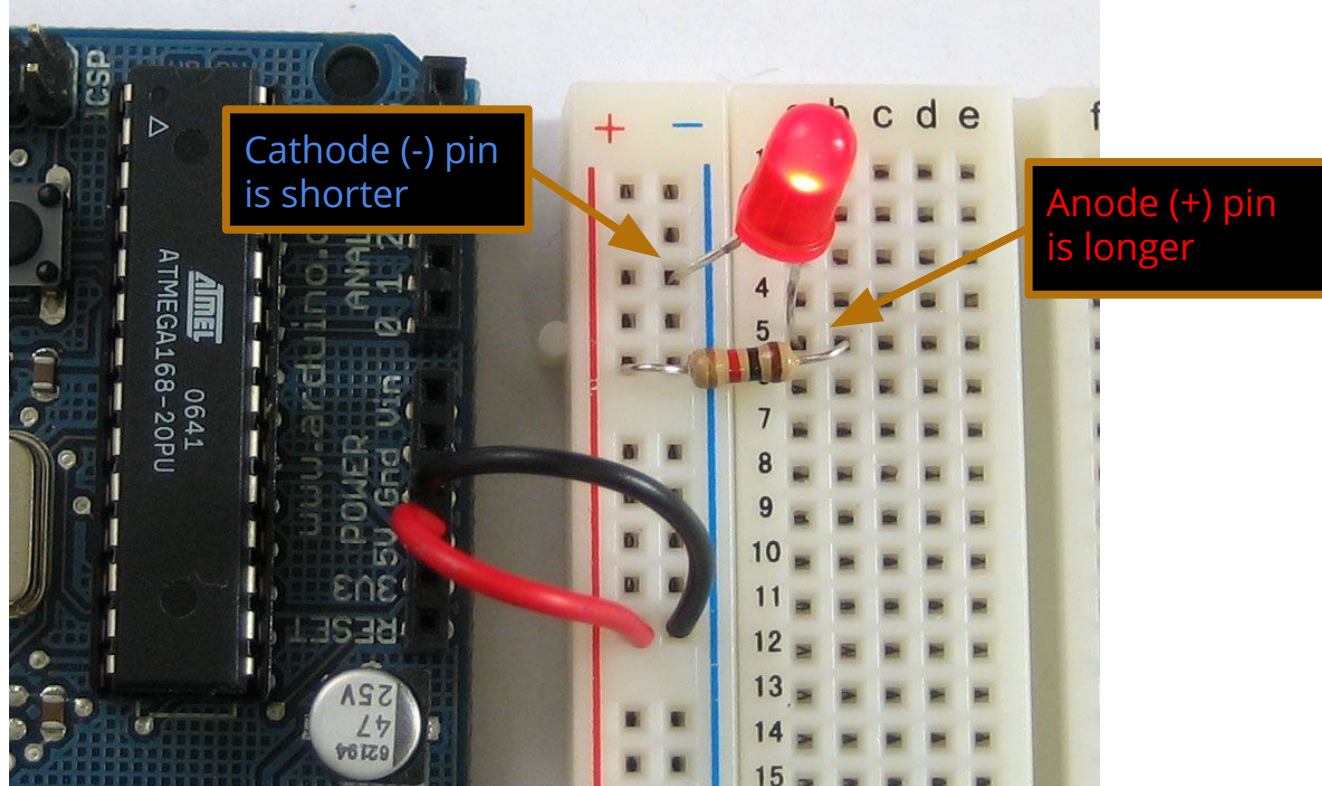


# Breadboards Explained



Tip: It is good practice to have your power input connected to the red/positive rail and your ground pin connected to the blue/negative rail.

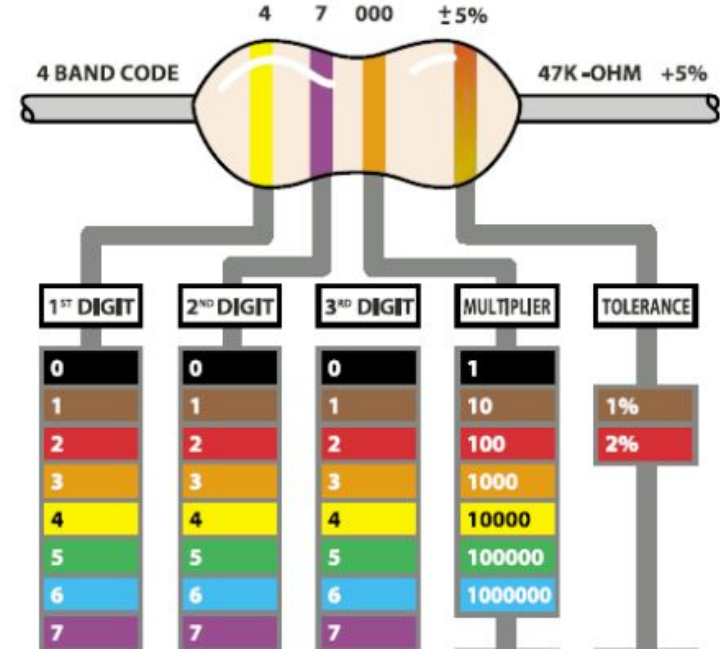
# LEDs



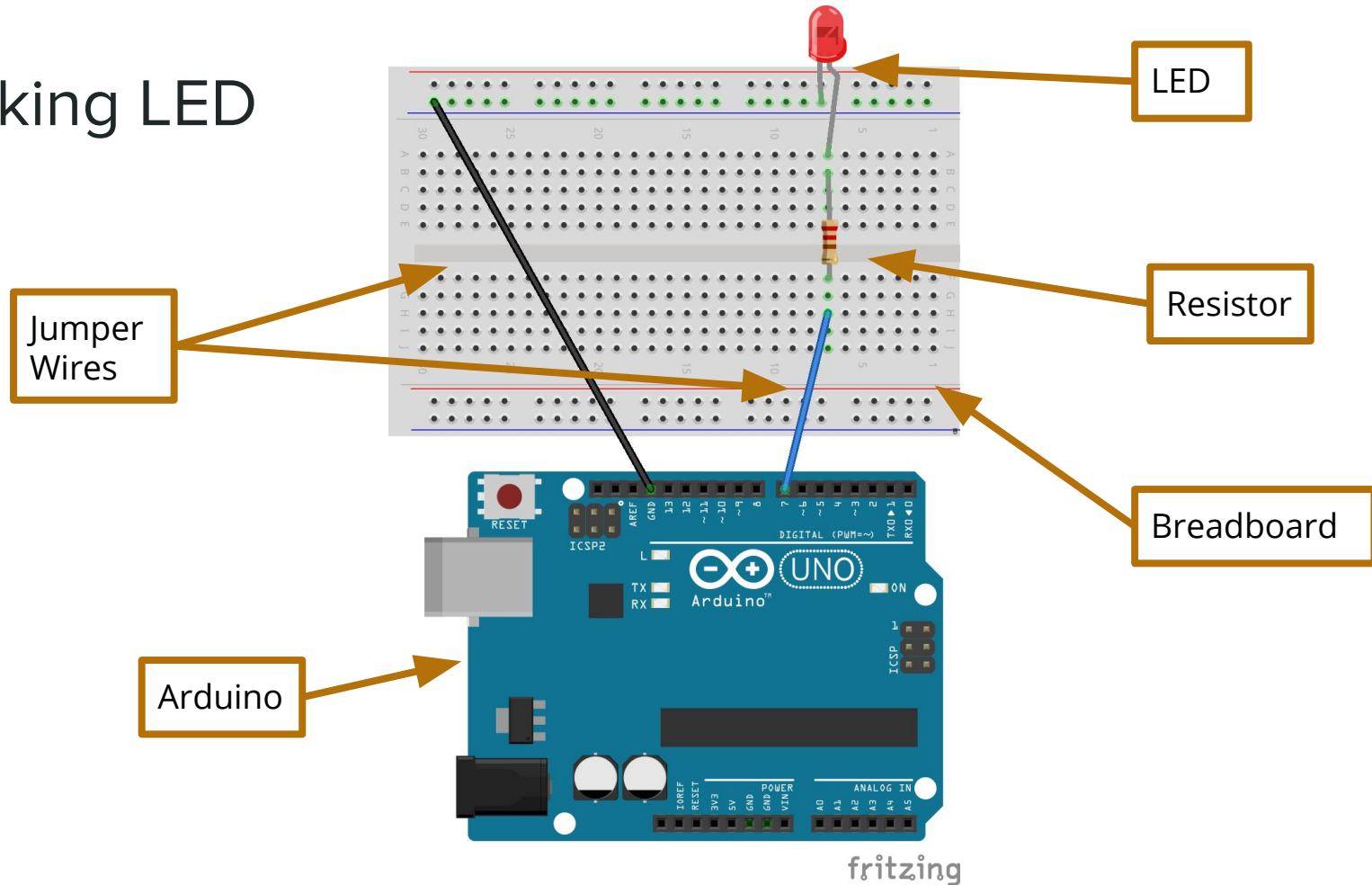
NOTE: Make sure the power input is connected to the Anode, and the ground pin is connected to the Cathode. Make sure you also have a resistor between either the power input and Anode, or the Cathode and ground pin. Failing to do either of these things can damage the LED or the Arduino.

# Resistors

- Resistors slow the electric current, and control where and how fast the current flows
- Resistance value is measured in ohms  $\Omega$ , which is represented by colored stripes on the body of the resistor
- Each stripe has a different value depending on the color and location as shown in the reference chart
- A potentiometer is a variable resistor

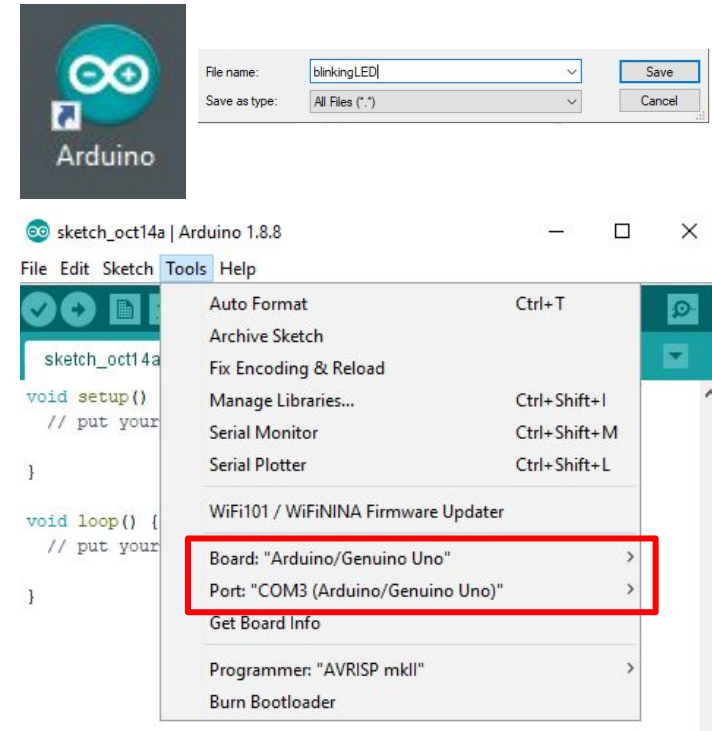


# Blinking LED



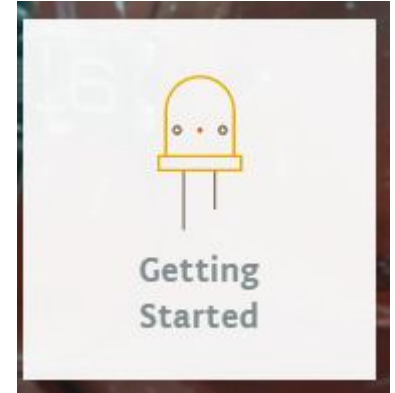
# Setting up Arduino

- Find and open Arduino on your desktop
- Click “File” in the top left corner and click save
- Save this tab as “blinkingLED”
- Connect the USB cord in your kit to the Arduino and the computer (USB port is on the left side of the monitor)
- Open the “Tools” Window and make sure the board has been recognized and the port is “COM3” with the name of the board after it




# Setting up Arduino (Online Web Editor)

- Search up create.arduino.cc
- Click Getting Started
- Scroll all the way down and click “Set up the Arduino Plugin”
- Click Next and follow the steps to download the plugin



# Setting up Arduino



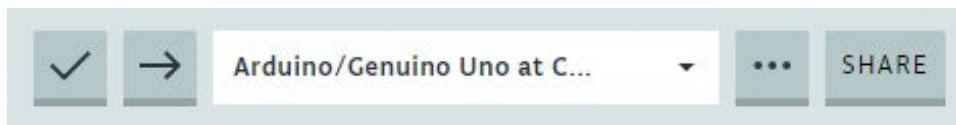
- If it doesn't automatically bring you to the login screen, click the 9 dots in the upper left hand corner and click Arduino Web Editor.
  - Click the Sign in with Google button
  - Sign in with your Google account
-  You must use a personal email!



The screenshot shows the login and account creation interface. On the left, under the heading 'LOGIN', there are two input fields: 'Username or e-mail' with the placeholder text 'USERNAME OR E-MAIL', and 'Password' with the placeholder text 'PASSWORD'. Below these fields is a link that says 'Forgot your username/password?'. At the bottom of the login section is a teal button labeled 'LOGIN'. On the right, under the heading 'CREATE A NEW ACCOUNT', there is a circular icon representing a user profile. At the bottom of the entire form, there is a button labeled 'Sign in with Google' which is highlighted with a red rectangular border. The Google 'G' logo is to the left of the text.

# Getting Started

- Click “NEW SKETCH” in the top left corner
- Click on the sketch name and rename it “digitalInput”
- Connect the USB cord in your kit to the Arduino and the computer
- The type of Arduino should have been recognized. If not, please type in chat.





# Blinking LED Code

led is a variable of the data type **int**, meaning it is a integer value.

This corresponds to the **pin number** on the Arduino.

To test your code, click the checkmark then the arrow!



```
blinkingLEDDemo $  
  
int led = 7;  
void setup() {  
  // put your setup code here, to run once:  
  pinMode(led, OUTPUT);  
}  
  
void loop() {  
  // put your main code here, to run repeatedly:  
  digitalWrite(led, HIGH);  
  delay(1000);  
  digitalWrite(led, LOW);  
  delay(1000);  
}
```

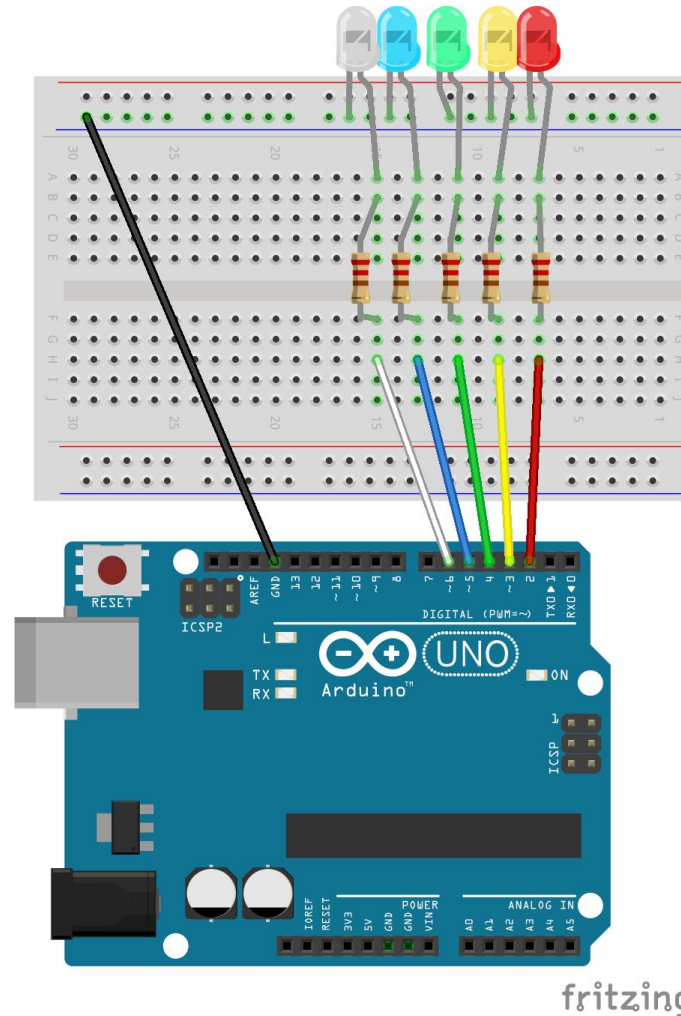
pinMode is a **function** that takes two parameters, led and OUTPUT. OUTPUT sets it so the LED is able to blink.

digitalWrite is a **function** that tells the LED to be in an "on" state (HIGH) or an "off" state (LOW)

# More LEDs

## Parts:

- Arduino
- Breadboard
- Jumper Wires
- Resistor
- LED




## Code Setup

- Click “File” in the top left corner and click save
- Click “File” again and open a “New” sketch
- Save this as “flowingLED”

# Line LED Code

lowestPin and highestPin are both integers. The term "const" means this value is constant and cannot be changed.



lineLEDdemo

```
/*  
*****  
const int lowestPin = 2;//the lowest one attach to  
const int highestPin = 7;//the highest one attach to  
*****  
*/  
void setup()  
{  
  //set pins 2 through 7 as output  
  for(int thisPin = lowestPin;thisPin <= highestPin;thisPin++)  
  {  
    pinMode(thisPin,OUTPUT); //initialize thisPin as an output  
  }  
}
```

This for-loop iterates through the pins from **lowest** value to **highest** value pin and turns them **on**, waits .1 seconds, then turns it off.

```
void loop()
{
    //iterate over the pins
    //turn the led on from lowest to the highest
    for(int thisPin = lowestPin;thisPin <= highestPin;thisPin++)
    {
        digitalWrite(thisPin,HIGH); //turn this led on
        delay(100); //wait for 100 microseconds
        digitalWrite(thisPin, LOW); //turn this led off
    }

    //turn the led on from highest to lowest
    for(int thisPin = highestPin;thisPin >= lowestPin;thisPin--)
    {
        digitalWrite(thisPin,HIGH); //turn this led on
        delay(100); //wait for 100 microseconds
        digitalWrite(thisPin,LOW); //turn this led off
    }
}
```

This for-loop iterates through the pins from **highest** value to **lowest** value pin and turns them **on**, waits .1 seconds, then turns it off.

[Click here for the repl.it file!](#)

# Flowing LED Code

lowestPin and highestPin are both integers. The term "const" means this value is constant and cannot be changed.

flowingLEDdemo

```
/*
*****
const int lowestPin = 2;//the lowest one attach to
const int highestPin = 7;//the highest one attach to
*****
void setup()
{
    //set pins 2 through 7 as output
    for(int thisPin = lowestPin;thisPin <= highestPin;thisPin++)
    {
        pinMode(thisPin,OUTPUT); //initialize thisPin as an output
    }
}
```

This for-loop iterates through the pins from **lowest** value to **highest** value pin and turns them **on**.

This for-loop iterates through the pins from **highest** value to **lowest** value pin and turns them **on**.

[Click here for the repl.it file!](#)

```
void loop()
{
    //iterate over the pins
    //turn the led on from lowest to the highest
    for(int thisPin = lowestPin;thisPin <= highestPin;thisPin++)
    {
        digitalWrite(thisPin,HIGH);//turn this led on
        delay(100);//wait for 100 microseconds
    }
    //fade from the highest to the lowest
    for(int thisPin = highestPin;thisPin>=lowestPin;thisPin--)
    {
        digitalWrite(thisPin,LOW);//turn this led off
        delay(100);//wait for 100 microseconds
    }

    //turn the led on from highest to lowest
    for(int thisPin = highestPin;thisPin>=lowestPin;thisPin--)
    {
        digitalWrite(thisPin,HIGH);//turn this led on
        delay(100);//wait for 100 microseconds
    }
    //fade from lowest to highest
    for(int thisPin = lowestPin;thisPin <= highestPin;thisPin++)
    {
        digitalWrite(thisPin,LOW);//turn this led off
        delay(100);//wait for 100 microseconds
    }
}
```

This for-loop iterates through the pins from **highest** value to the **lowest** value pin and turns them **off**.

This for-loop iterates through the pins from **lowest** value to the **highest** value pin and turns them **off**.

# Mystery Pattern

What do you think the code shown on the right does?

multi\_LED\_mystery

```
/*
*****
const int lowestPin = 2; //the lowest pin your LEDs have been attached to
const int highestPin = 7; //the highest pin your LEDs have been attached to
*****
*/

void setup()
{
  //set pins 2 through 7 as output
  for(int thisPin = lowestPin; thisPin <= highestPin; thisPin++)
  {
    pinMode(thisPin, OUTPUT); //initialize thisPin as an output
  }
}

void loop()
{
  //iterate over the pins
  //turn the led on from lowest to the highest
  for(int x = 2; x <= 3; x++){
    digitalWrite(x, HIGH);
    digitalWrite(x+2, HIGH);
    digitalWrite(x+4, HIGH);
    delay(200);
    digitalWrite(x, LOW);
    digitalWrite(x+2, LOW);
    digitalWrite(x+4, LOW);
  }
}
```

# Thank you!

---

CCA Girls Who Code

[ccagirlswhocode@gmail.com](mailto:ccagirlswhocode@gmail.com)

[www.ccagirlswhocode.weebly.com](http://www.ccagirlswhocode.weebly.com)



# Production Team

Program Director: Samantha Prestrelski

Head Instructor: Stefan Prestrelski

Teaching Assistant: Sarah Luo