



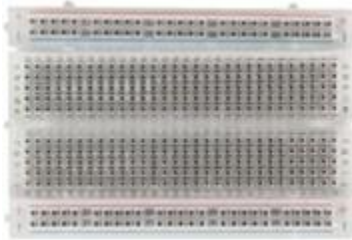
This entry level kit contains the basics you'll need to get started in the world of Arduino & features projects that set the foundations for putting your own ideas into action.

What is Arduino

Arduino refers to an open-source electronics platform or board and the software used to program it. Arduino is designed to make electronics more accessible to artists, designers, hobbyists and anyone interested in creating interactive objects or environments.

What's Included

Image	Part	Description
	Uno Board	The duinotech Uno is an open source single-board controller capable of controlling multiple input and output devices.
	USB Cable	You'll need this to connect your Uno to your computer to power and load code onto the board.

Image**Part****Description**

Breadboard

Breadboards are handy boards used for testing circuit designs. With multiple pins you can easily connect multiple sensors, switches and LEDs.



Jumper Leads

Jumper wires are wires that have connector pins at either end, and are commonly used between the Uno, breadboard & and any external sensors you might be using in your next project.



LEDs

LED stands for 'Light Emitting Diode', a semiconductor device that converts

Image

Part

Description



Tactile
Switches

electricity into light. In simple terms, it's a small light that operates on a low voltage.

These switches can be used to turn on and off circuits. They are momentary, meaning they only operate when the button is held down.



DC Motor
with Fan

DC motor is one type of motor that uses the DC current to convert electrical energy into mechanical energy. This small motor & fan combination is used in project 4 to control the

Image**Part****Description**

Resistors


speed of the fan.

A resistor is an electrical component that limits or regulates the flow of electrical current in an electronic circuit.



Potentiometer

A potentiometer is a three-terminal resistor with a sliding or rotating contact that forms an adjustable voltage divider. Where only two pins are used, it becomes a variable resistor.

Image	Part	Description
	Buzzer	A buzzer is an electrical device that makes a buzzing noise and is used for signaling.

Getting Started

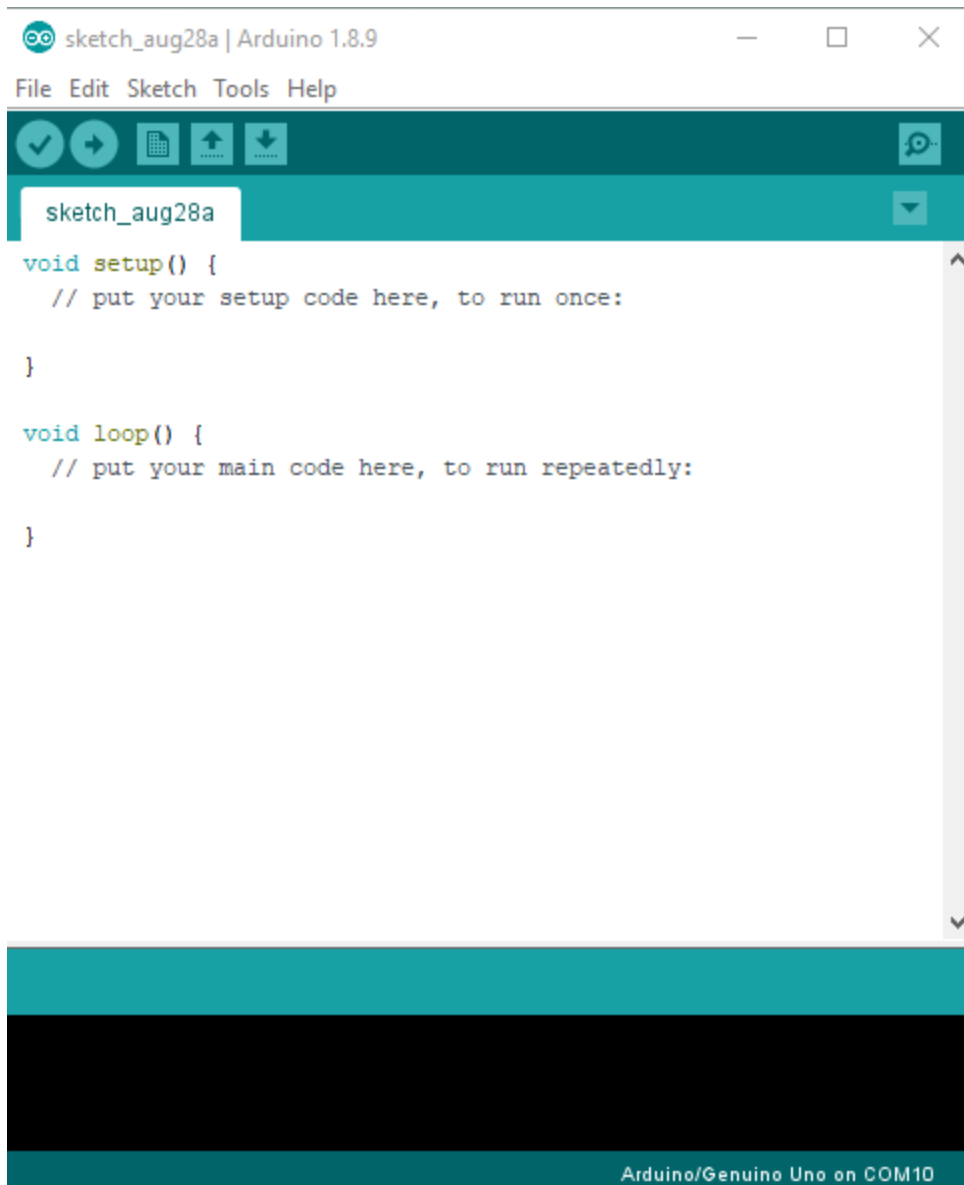
Install Arduino IDE

The Arduino IDE can be found on the official Arduino website at <https://www.arduino.cc/en/Main/Software>

Download the version suitable for your computer and follow the prompts to install the software package.

Write or Paste Code

Opening the Arduino IDE for the first time, you will be presented with the following screen.



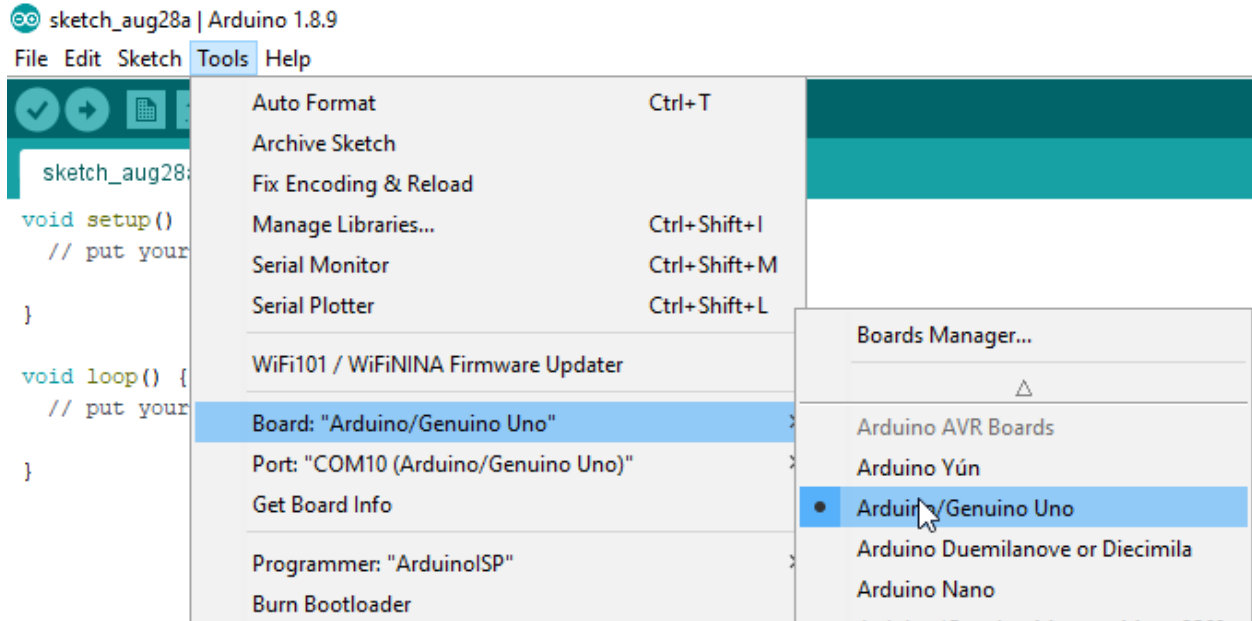
This is where you will write or paste in the code for your project.

Uploading Code

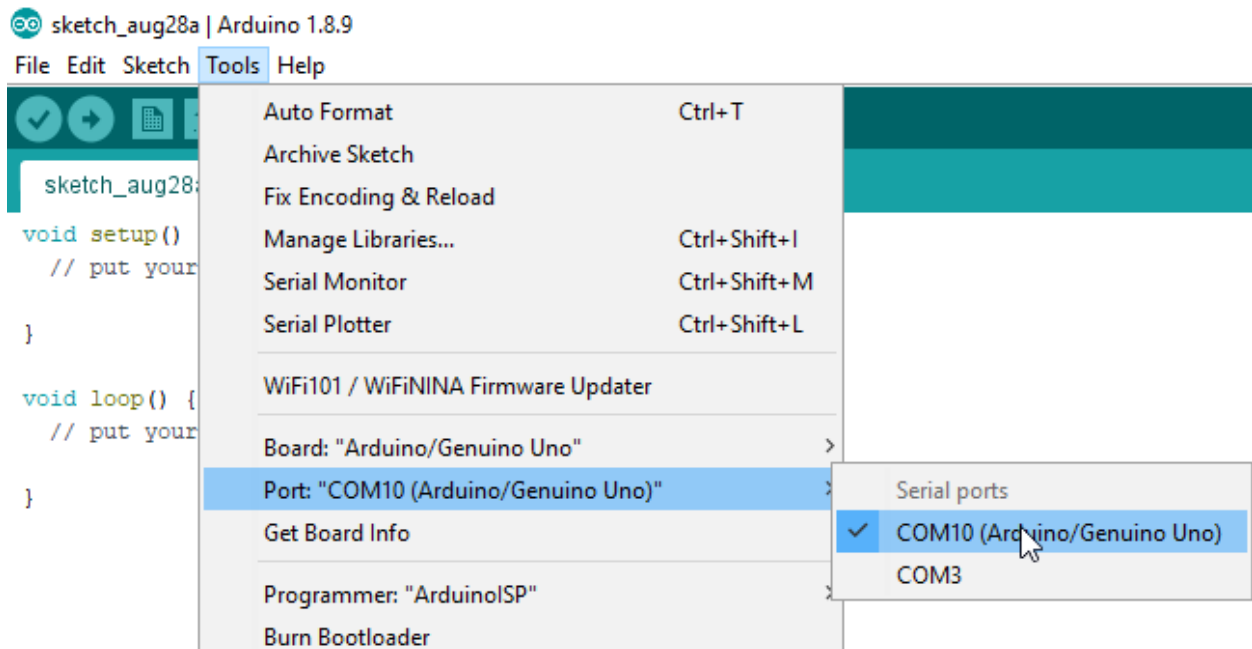
Now that you have prepared your code, it's time to upload it onto the board.

Selecting Board type & Port

You'll need to select the entry in the Tools > Board menu that corresponds to your Arduino board. For the duinotech Uno, you will select Arduino/Genuino Uno.

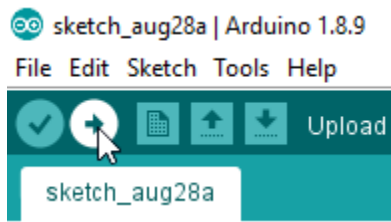


Select the serial device of the board from the Tools | Serial Port menu. This is likely to be COM3 or higher (COM1 and COM2 are usually reserved for hardware serial ports). To find out, you can disconnect your board and re-open the menu; the entry that disappears should be the duintech Uno board. Reconnect the board and select that serial port.



Uploading Code

Now, simply click the "Upload" button in the environment. Wait a few seconds - you should see the RX and TX LEDs on the board flashing. If the upload is successful, the message "Done uploading." will appear in the status bar.



The example projects in this kit are free of errors, however, if you write your own code you may find an error when uploading your code. The software will show a message indicating the type of error and the location in which it appears in the code.

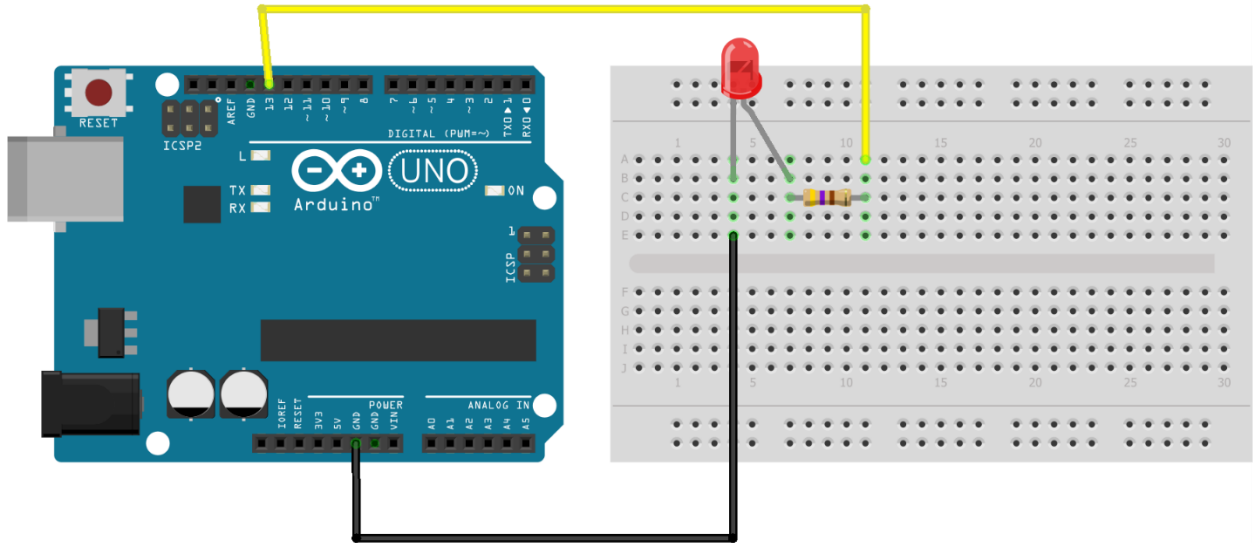
Projects

Now that you have the Arduino IDE setup and ready to use you can begin working through our beginner projects.

Project	Outcome
Using an LED	This basic project will introduce the beginner Arduino user to simple code & circuits by turning on & off an LED.
Using a Potentiometer	Use a potentiometer to adjust the brightness of the in-built LED on the Arduino Uno.
Using Buttons	Use feedback from an analogue sensor to activate a buzzer via the Arduino board.
Fan Speed Controller	Use variable feedback from a potentiometer to adjust the speed of the motor.
Traffic Lights	Simulate traffic lights using multiple LEDs & a loop circuit in Arduino.

Using an LED

This basic project will introduce the beginner Arduino user to simple code & circuits by turning on & off an LED.



fritzing

Required Kit Components

Part	Quantity
LED	1
Resistor	1
Jumper Wires	2

Code

```
void setup() // Runs once when sketch starts
{
  pinMode(13, OUTPUT); // Setting the LED pin as an output
}

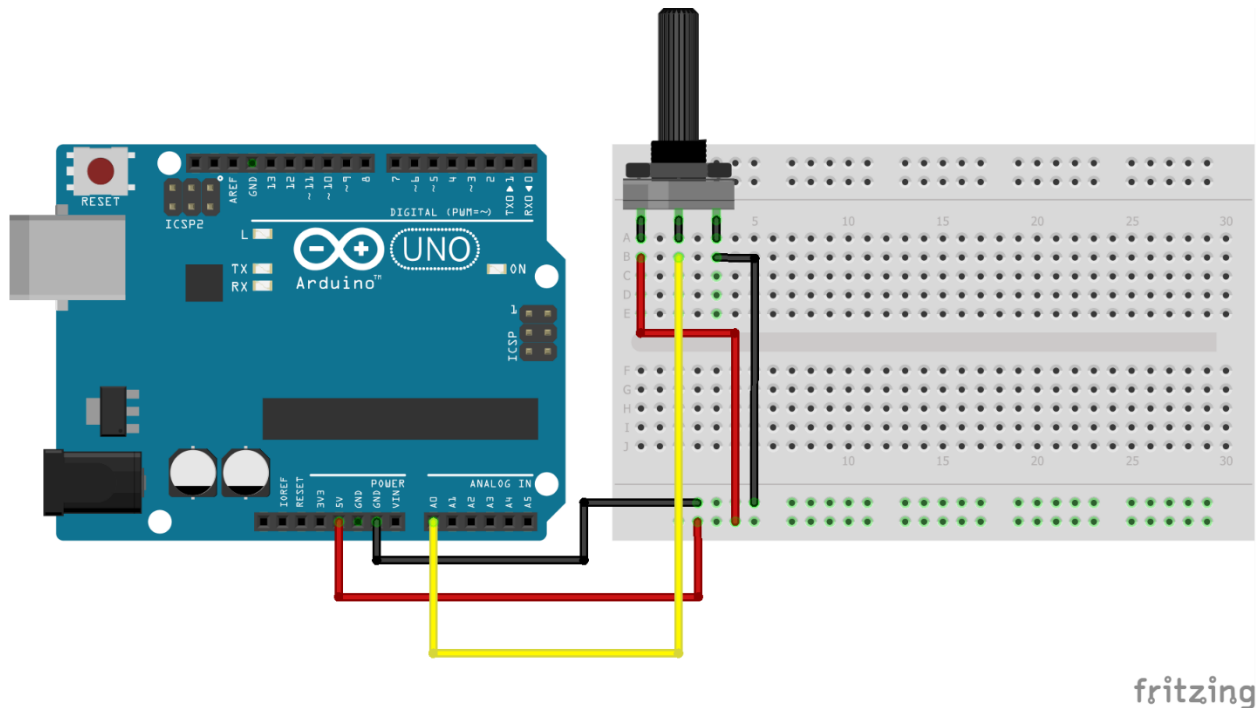
void loop() // Runs repeatedly
{
  digitalWrite(13, HIGH); // Turning the LED on
  delay(1000); // Waiting 1 second
  digitalWrite(13, LOW); // Turning the LED off
  delay(1000); // Waiting 1 second
}
```

Using a Potentiometer

Use a potentiometer to adjust the brightness of the in-built LED on the Arduino Uno.

This extends the previous project:

1. Change the LED connection from pin 13 to pin 9.
2. Connect the potentiometer to the circuit as below.



Required Kit Components

Part	Quantity
Potentiometer	1
Jumper Wires	6

Code

```
int potPin = A0;           // Input pin from the potentiometer
int ledPin = 9;           // Output pin to the LED
```

```

int potVal = 0;          // A variable to store the potentiometer value
int brightness = 0;

void setup()            // Runs once when sketch start
{
  pinMode(ledPin, OUTPUT);    // Setting the LED pin as an output
}

void loop()             // Runs repeatedly
{
  //read from the pot pin, and store it into potval
  potVal = analogRead(potPin);

  // "map" the brightness, so that potval (0->1023)
  // corresponds with a value between (0->255)

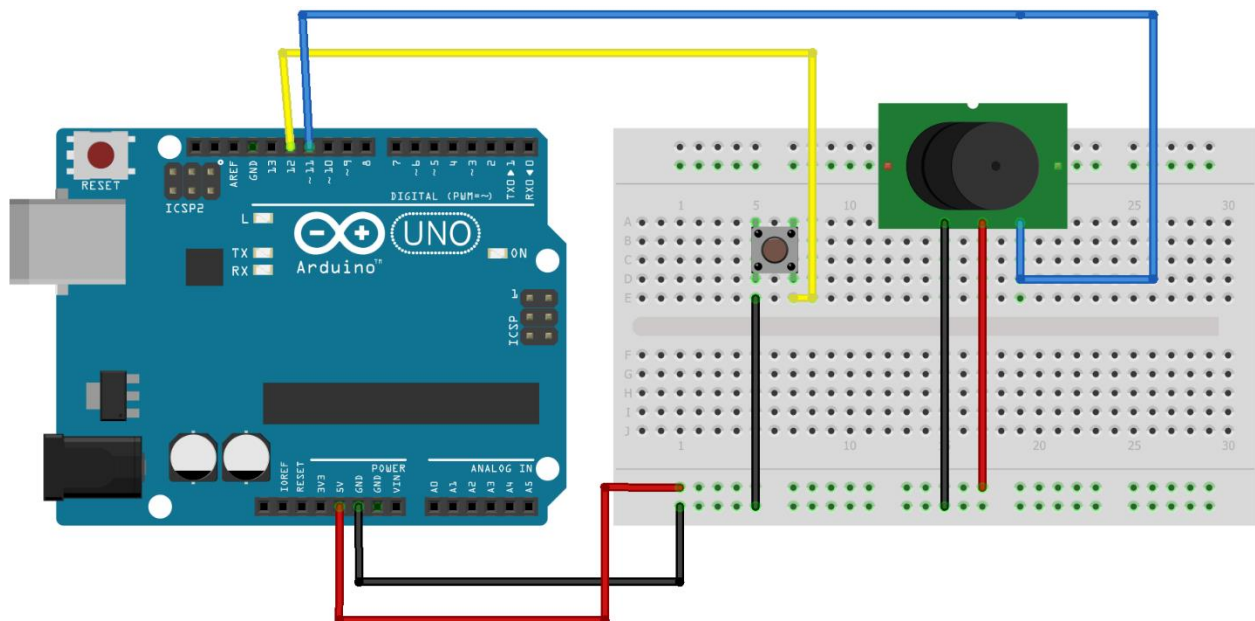
  brightness = map(potVal , 0, 1023, 0, 255);

  //analog write uses PWM to control the brightness of the LED.
  // it will only work on the PWM pins, which pin 9 is.
  analogWrite(ledPin, brightness);
}

```

Using Buttons

Use feedback from an analogue sensor to activate a buzzer via the Arduino board.



fritzing

Required Kit Components

Part	Quantity
Switch	1
Buzzer	1
Jumper Wires	7

Code

```

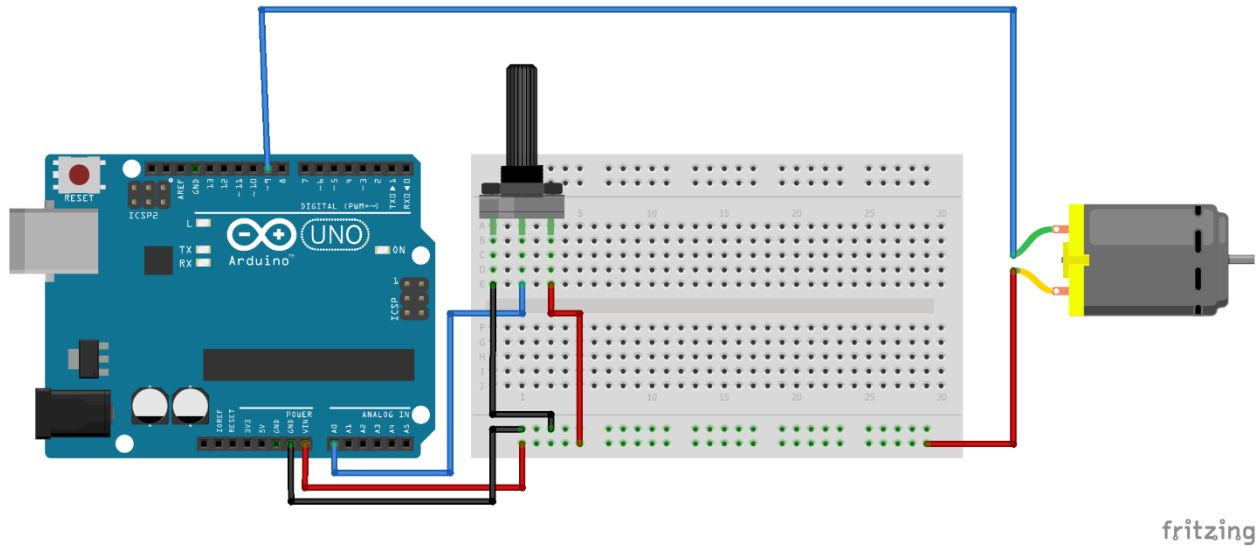
// Setting variables which can be
easily called to later
  int Button = 12;           // Input pin from the button
  int Buzzer = 11;          // Output pin to the buzzer
  int Val = 0;              // A variable to store the button
value
void setup()                // Runs once when sketch starts
                             // Setting the pin type & defining
the I/O
{
  pinMode(Button, INPUT_PULLUP); // Setting the button pin as an input which
                             // uses an internal pullup resistor
  on the Uno board
  pinMode(Buzzer, OUTPUT);     // Setting the buzzer pin as an
output
}

void loop()                 // Runs repeatedly
{
  Val = digitalRead(Button); // Setting the Val variable to the
output of the
                             // button, which can be either HIGH
or LOW
  if (Val == LOW) {         // Statement to determine the state
of the button
    digitalWrite(Buzzer, HIGH); // If the button is pressed, the buzzer
will sound
  } else {
    digitalWrite(Buzzer, LOW);  // Else, the buzzer will not sound
  }
}

```

Fan Speed Controller

Use variable feedback from a potentiometer to adjust the speed of the motor.



Required Kit Components

Part	Quantity
Potentiometer	1
Motor	1
Jumper Wires	5

Code

```
easily called to later
int POT_PIN = A0;
int MOTOR_PIN = 9;
int motorSpeed = 0;
speed value
int potVal = 0;
potentiometer value

void setup()

// Setting variables which can be
// Input pin from the potentiometer
// Output pin to the motor
// A variable to store the motor
// A variable to store the
// Runs once when sketch starts
```

```

{
the I/O
  pinMode(MOTOR_PIN, OUTPUT);
output
}

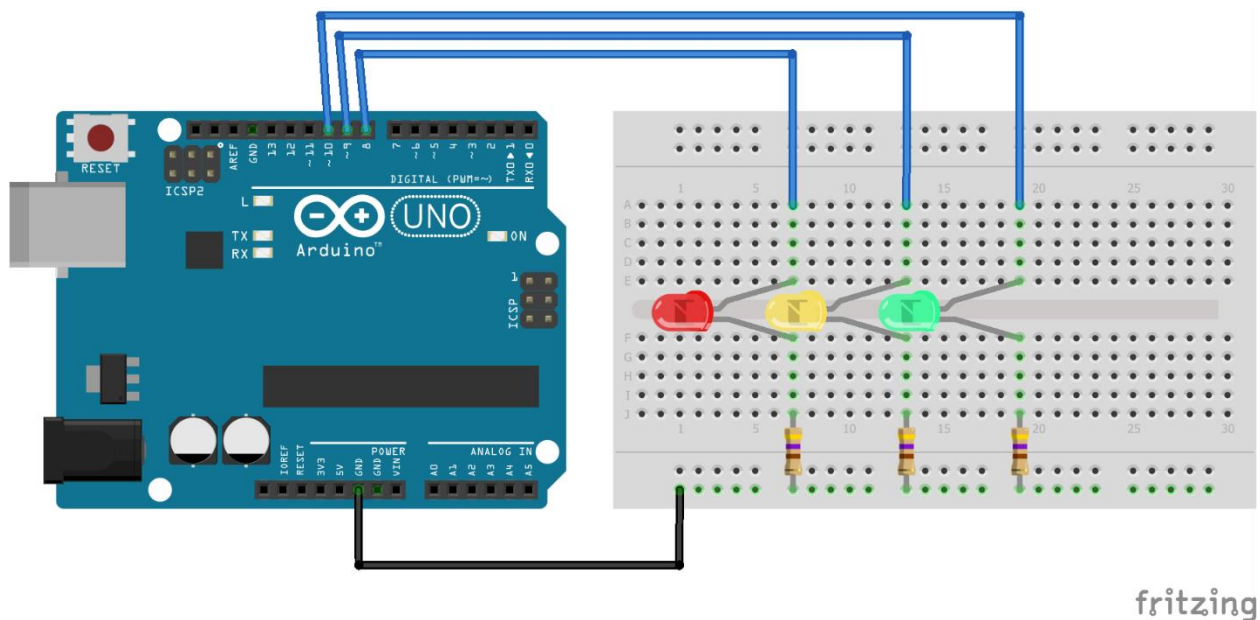
void loop()
{
  potVal = analogRead(POT_PIN);
the reading from the POT_PIN
  motorSpeed = map(potVal, 0, 1023, 0, 255);
to an equivalent variable

potVal which is between 0 & 1023
  analogWrite(MOTOR_PIN, motorSpeed);
motorSpeed pin
}
// Setting the pin type & defining
// Setting the motor pin as an
// Runs repeatedly
// Setting the potVal variable to
// Setting the motorSpeed variable
// between 0 & 255 based off the
// Writing the mapped value to the

```

Traffic Lights

Simulate traffic lights using multiple LEDs & a loop circuit in Arduino.



Required Kit Components

Part	Quantity
LEDs	3 (green/red/yellow)

Part	Quantity
Resistors	3
Jumper Wires	4

Code

```
// Setting variables which can be easily called to later
int GREEN = 8;           // Setting the pins from the LEDs
int YELLOW = 9;
int RED = 10;
int DELAY_GREEN = 5000; // Setting variables for delays that can be adjusted
int DELAY_YELLOW = 2000; // without altering each function
int DELAY_RED = 5000;

void setup()           // Runs once when sketch starts
{
  pinMode(GREEN, OUTPUT); // Setting the LEDs to outputs
  pinMode(YELLOW, OUTPUT);
  pinMode(RED, OUTPUT);
}

void loop()           // Runs repeatedly
{
  green_light();      // Running each function followed by a delay set by variables
  delay(DELAY_GREEN);
  yellow_light();
  delay(DELAY_YELLOW);
  red_light();
  delay(DELAY_RED);
}

void green_light()    // A function for turning only the green LED on
{
  digitalWrite(GREEN, HIGH); // Turning on the green LED & the others off
  digitalWrite(YELLOW, LOW);
  digitalWrite(RED, LOW);
}

void yellow_light()  // A function for turning only the yellow LED on
{
  digitalWrite(GREEN, LOW); // Turning on the yellow LED & the others off
  digitalWrite(YELLOW, HIGH);
  digitalWrite(RED, LOW);
}

void red_light()     // A function for turning only the red LED on
{
```

```
digitalWrite(GREEN, LOW);    // Turning on the red LED & the others off
digitalWrite(YELLOW, LOW);
digitalWrite(RED, HIGH);
}
```