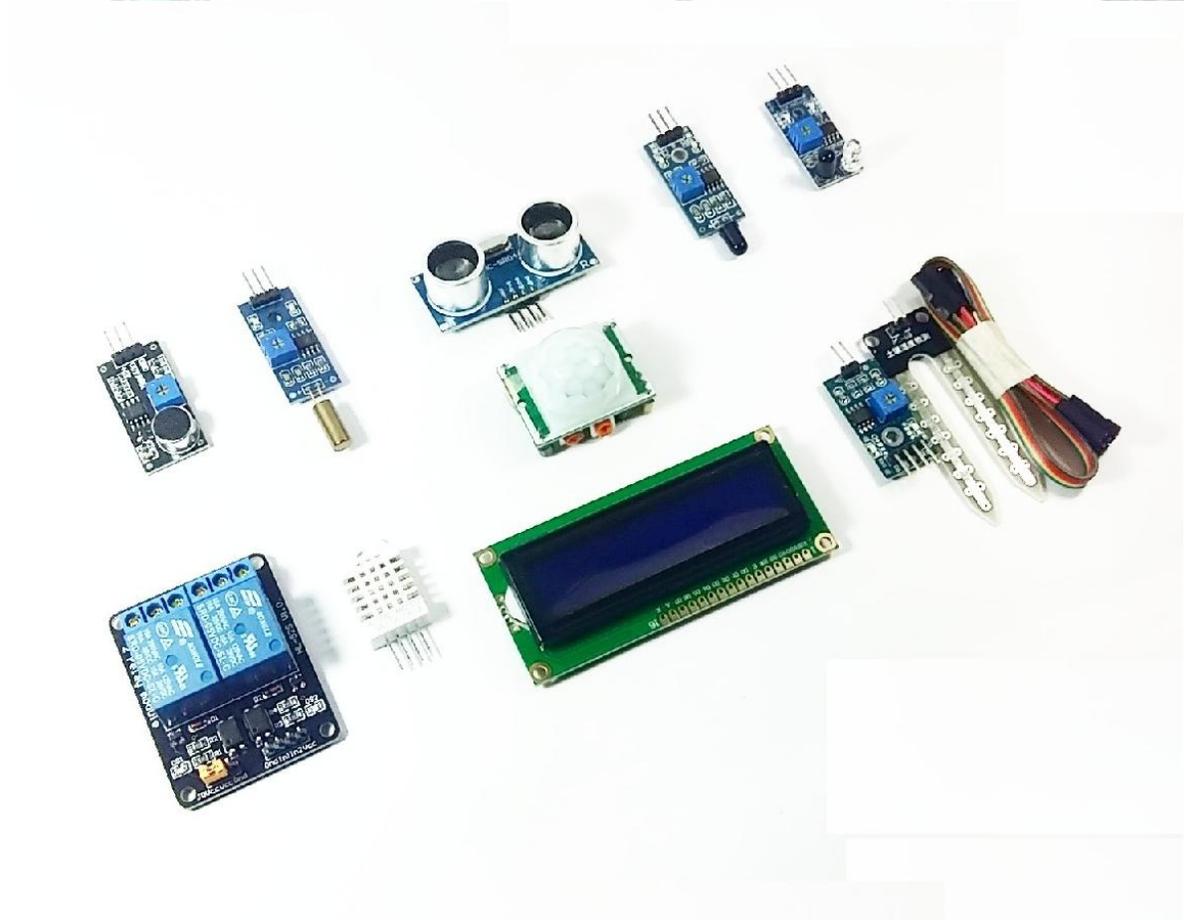


Sensors and Modules Kit

version 2.0
(Arduino Manual)



Sensors and modules are the bread and butter of the microcontroller. They allow your microcontroller to interact with its surrounding, be it to check for temperature, see, or respond. Your microcontroller becomes an interactive device able to interact with you and its environment. The Sensors and Modules Kit is a compilation of the most common sensors and modules used in the Arduino Environment. It provides you with a wide range of parameters and responses to do, from temperature, distance to displaying data and controlling an electronic device.

This manual will guide you into using the various sensors in the Arduino Environment, including adding three projects integrating two or more modules. Get your kits ready, and start the maker experience!

Arduino: A Brief Encounter

Arduino Hardware (Arduino UNO R3)



Arduino Software (Arduino C/C++ IDE Programming)

```

Blink | Arduino 1.6.9
File Edit Sketch Tools Help
Blink $
This example code is in the public domain.
modified 9 May 2014
by Scott Fitzgerald
*/

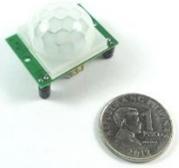
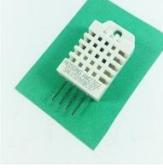
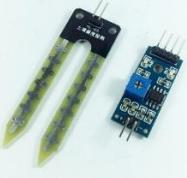
// the setup function runs once when you press reset or power the board
void setup() {
  // initialize digital pin 13 as an output.
  pinMode(13, OUTPUT);
}

// the loop function runs over and over again forever
void loop() {
  digitalWrite(13, HIGH); // turn the LED on (HIGH is the voltage level)
  delay(1000); // wait for a second
  digitalWrite(13, LOW); // turn the LED off by making the voltage LOW
  delay(1000); // wait for a second
}
    
```

Arduino is an easy-to-use tool for learning programming and electronics. The Arduino is an environment, with hardware through microcontroller doing the action being dictated by the software, which is the program. Through this interaction, you can control lights, monitor the environment, program motors or create your own robot. The Arduino environment has two main parts mainly the Arduino Board (Hardware) and Arduino IDE (Software)

The Arduino, when compared to other environments using microcontrollers, is easier to use, is well-documented supporting materials and constant support of other programmers and developers. Arduino also has partnerships with other organizations promoting development, especially on prototyping and do-it-yourself (DIY) systems and projects. It is also scalable in terms of scope, where you can start from just blinking an LED to creating your own 3D printer.

Parts List

 <p>PIR Motion Sensor</p>	 <p>Ultrasonic Sensor HC-SR04</p>	 <p>DHT-22 Temperature and Humidity Sensor</p>	 <p>Flame/Fire/Light Sensor</p>	 <p>Line Follower Sensor Board 1 channel</p>
 <p>Sound Sensor</p>	 <p>Soil Moisture Sensor</p>	 <p>Tilt Sensor</p>	 <p>LCD 2x16 Module</p>	 <p>Relay Module – 2 Channel</p>

For the entire manual, you will also need the additional parts:

 <p>Arduino Uno R3</p>	 <p>USB A to USB B Cable</p>	 <p>Breadboard (Full-Size)</p>	 <p>Connecting Wires – Male-Male 40 pcs</p>
 <p>Connecting Wires – Male-Female 40 pcs</p>	 <p>Light Emitting Diode – Red (1 pc) Green (1 pc)</p>	 <p>Resistor – 330 Ω (1 pc) 10 kΩ (1 pc)</p>	 <p>Potentiometer – 10 kΩ (1 pc)</p>

PIR Motion Sensor HC-SR501



The PIR Motion Sensor is an affordable sensor for commonly used for motion detection. This is possible by detecting changes in the infrared levels in the environment, a common scene when there is motion in the environment. The sensor has only three pins for usage, making it a good starting sensor for electronic enthusiasts and beginners.

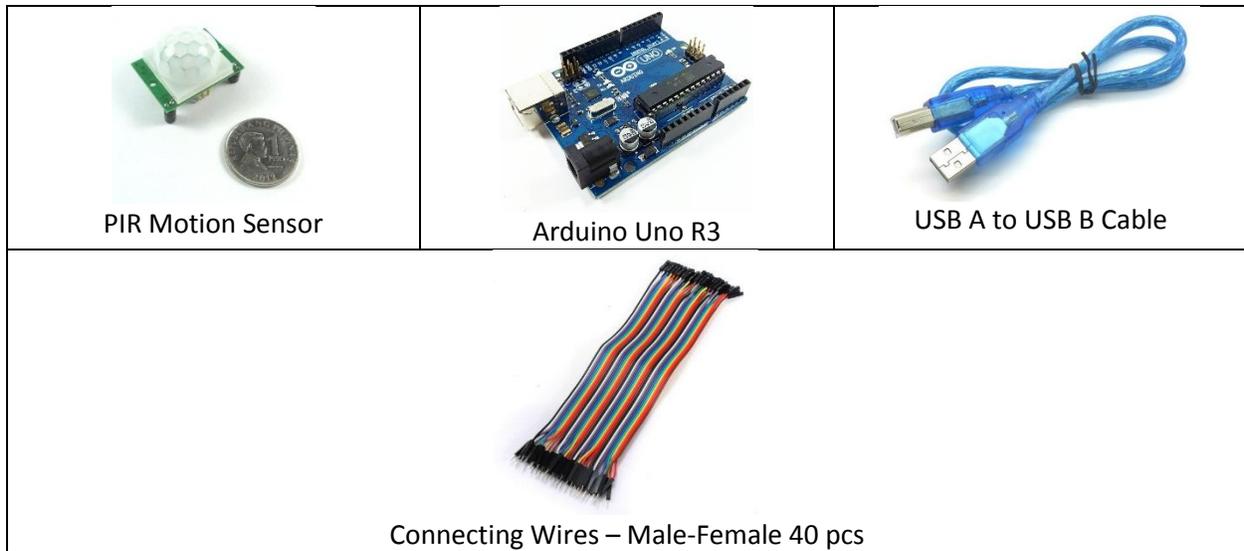
The PIR sensor only gives out HIGH or LOW signal output, which means that no serial communication is needed. Simply plug to your microcontroller, read the data and create a response for the two conditions.

HARDWARE SPECIFICATIONS

- Working Voltage: DC 5V -15V
- Signal Output: 0 (no movement), 3.3V (movement detected)
- Delay Time: Adjustable (18 – 300 seconds)
- Lock time: 0.2 seconds
- Trigger Methods: L- disable repeat trigger, H – enable repeat trigger
- Sensing range: less than 120°, within 7 meters
- Dimension: 32*24 mm

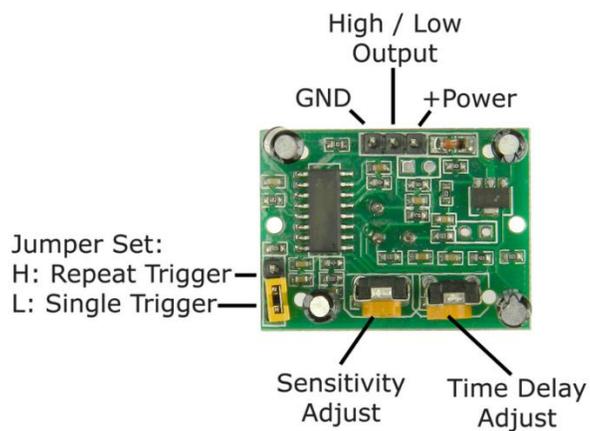
PARTS LIST

For this setup, we will need the following materials:



HARDWARE OVERVIEW

The line follower sensor has three pins: VCC, GND and Out.



INPUT	Description
GND	To be connected to the GND pin in a microcontroller.
VCC	Supplies power to the module. Could be connected to a +5V pin.
OUTPUT	
Out	Sends a high or low pulse signal to indicate whether there is a black or a white color detected.

The PIR has two detection modes: single cycle mode and retriggerable mode. The single cycle mode (L) detects a single movement only, and stays on as set in by the time delay adjust knob. Any movement done after trigger on single cycle mode is not detected by the PIR sensor. On retriggerable mode, any movement done after trigger resets the time the sensor is on. To set to

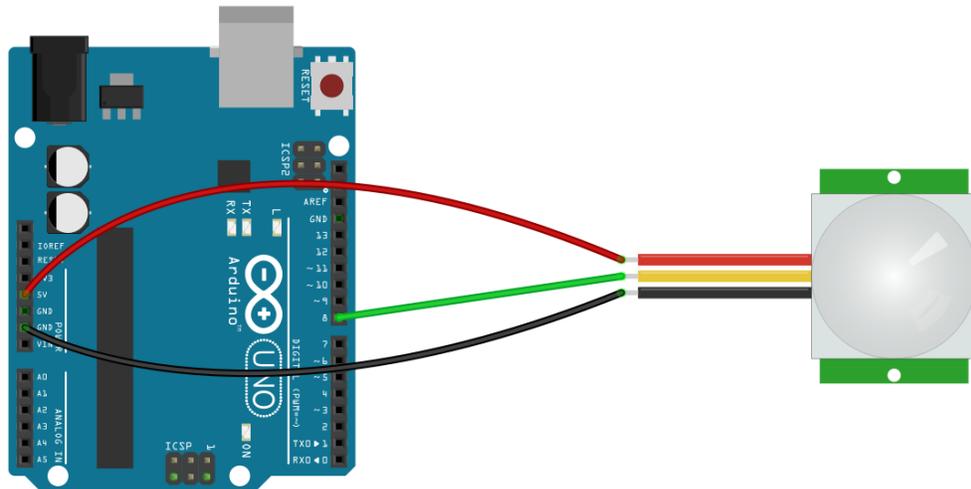
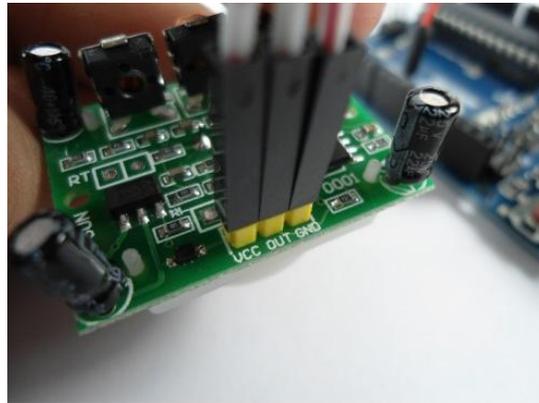
retriggerable mode, move the jumper header to repeat trigger pin. For single cycle trigger, move header to single trigger pin.

To adjust the duration of the ON-period of the PIR, rotating the potentiometer clockwise increases the delay by 5 minutes, while rotating counterclockwise decreases it to 18 seconds. Sensitivity increases up to 7 meters on clockwise rotation, and up to 3 meters, counterclockwise.

WIRING CONNECTION

Connect the line sensor to the Arduino Board according to the table below:

<u>Line/Obstacle Sensor</u>	--->	<u>Arduino</u>
VCC	--->	5V
GND	--->	GND
OUT	--->	DIGITAL PIN #8



fritzing

ARDUINO CODE

On a blank sketch in the Arduino IDE program, code the code below:

```
/*test program for HC-SR501 Motion Sensor*/

int PIRpin = 8; // input pin for arduino to read motion sensor logic high data
int val;

void setup()
{
  Serial.begin(9600);
  pinMode(13, OUTPUT);
}

void loop()
{
  val = digitalRead(PIRpin);
  if (val == LOW)
  {
    Serial.println("No Motion");
    digitalWrite(13, LOW);
  }
  else
  {
    Serial.println("Motion Detected!");
    digitalWrite(13, HIGH);
  }
  delay(1000);
}
```

Compile the code, and upload it into the Arduino Board. Open Serial Monitor, and set baud rate to **“9600, BOTH NL & CR”**.

OUTPUT

After uploading the code open your Arduino Serial Terminal. It will display if motion is detected or not.

By placing and waving your hand or “body” in front of the motion sensor (with around 5 meter distance range in front of the sensor) the LED on the Arduino board will light up and in the Arduino Serial Terminal a “Motion Detected!” message will show. If no hand or body figure is moving in front of the sensor the LED on the Arduino will not light up and the Serial Terminal will display a “No Motion” message.

Ultrasonic Range, Distance and Proximity Sensor HC-SR04



The ultrasonic sensor is a sensor that senses objects in front of it using sound. It detects the distance of the closest object in front of the sensor from 2 cm up to 400 cm. This is useful for obstacle avoidance for proximity sensing systems.

HARDWARE SPECIFICATIONS

- Input voltage: 5V
- Input Current: 15 mA
- Detection Range: 0.02-4 m
- Angle Detection: 15 degrees

PARTS LIST

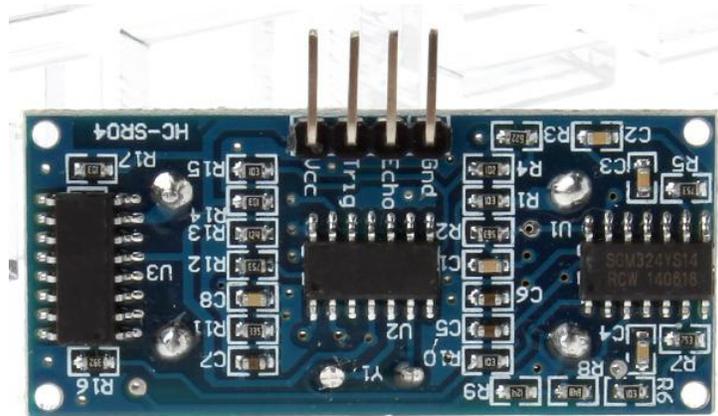
For this setup, we will need the following materials:

		
Ultrasonic Sensor HC-SR04	Arduino Uno R3	USB A to USB B Cable



HARDWARE OVERVIEW

The ultrasonic sensor has four pins to be connected to the microcontroller: VCC, GND, Trig and Echo.



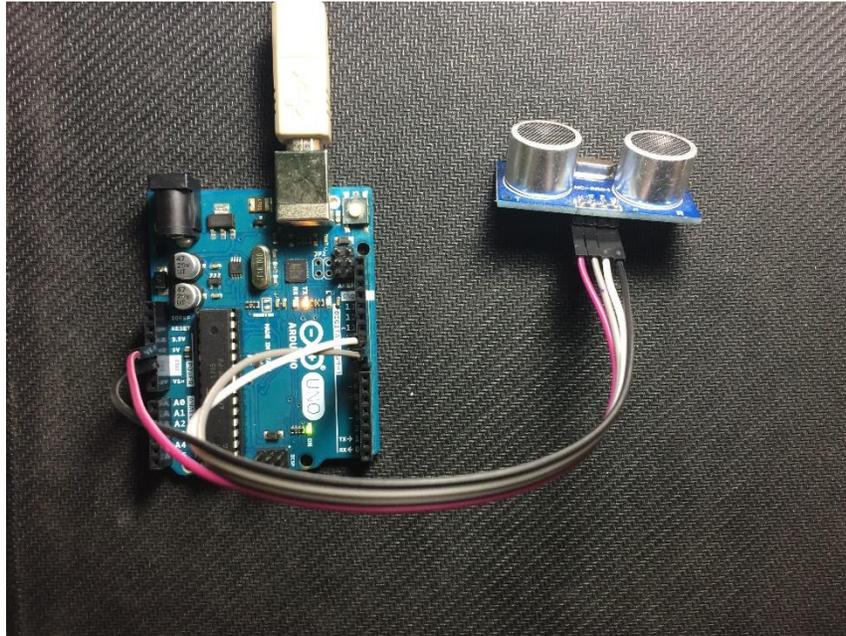
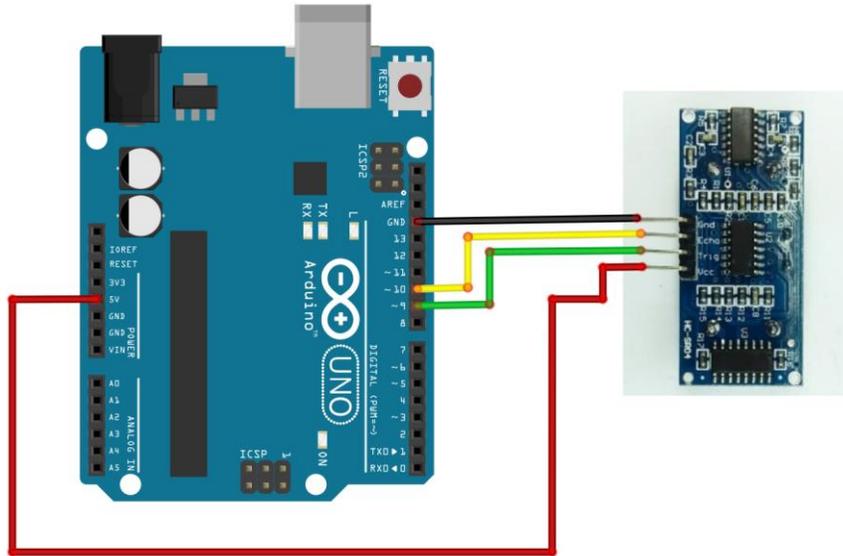
The table below describes the function of each pin in the module

INPUT	Description
GND	To be connected to the GND pin in a microcontroller.
VCC	Supplies power to the module. Could be connected to a +5V pin.
Trig	Sends pulse signal for the sensor to emit ultrasonic waves
OUTPUT	
Echo	Sends received sound wave data to microcontroller.

WIRING CONNECTION

Setup the circuit as shown below:

ULTRASONIC RANGE, DISTANCE AND PROXIMITY SENSOR



Ultrasonic Sensor Pin	Arduino Uno Pin
Vcc	5V (PURPLE WIRE)
Trig	Digital 9 (GRAY WIRE)
Echo	Digital 10 (WHITE WIRE)
GND	GND (BLACK WIRE)

ARDUINO CODE

Open Arduino IDE. Copy the code below.

/*

```

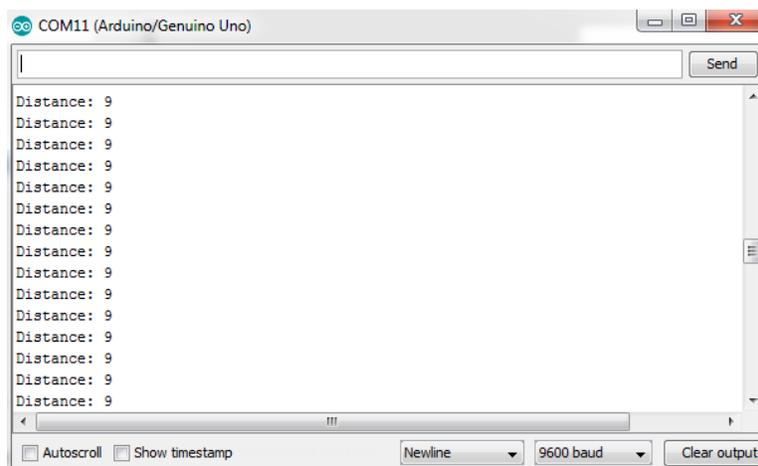
* Ultrasonic Sensor HC-SR04 interfacing with Arduino.
*/
// defining the pins
const int trigPin = 9;
const int echoPin = 10;
// defining variables
long duration;
int distance;
void setup() {
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  Serial.begin(9600); // Starts the serial communication
}
void loop() {
  // Clears the trigPin
  digitalWrite(trigPin, LOW);
  delayMicroseconds(2);
  // Sets the trigPin on HIGH state for 10 micro seconds
  digitalWrite(trigPin, HIGH);
  delayMicroseconds(10);
  digitalWrite(trigPin, LOW);
  // Reads the echoPin, returns the sound wave travel time in microseconds
  duration = pulseIn(echoPin, HIGH);
  // Calculating the distance
  distance= duration*0.034/2; //speed travels at 340 m/s or 0.034 cm/us, then sound
  bounces back so distance should be divided into two.
  // Prints the distance on the Serial Monitor
  Serial.print("Distance: ");
  Serial.println(distance);
}

```

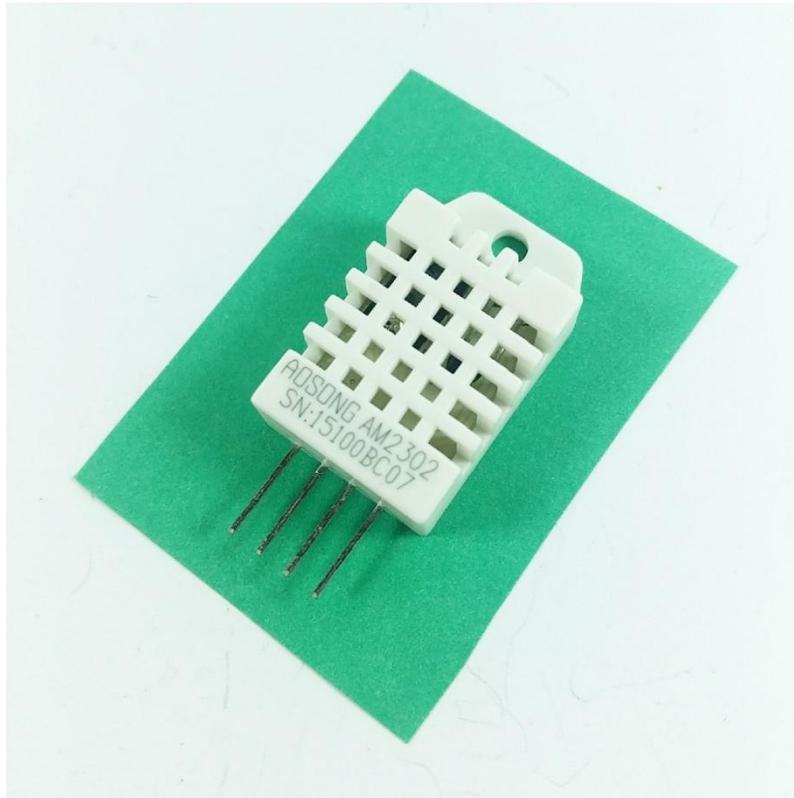
Upload the code. Open Serial Monitor and set Baud Rate to **“9600, Both NL & CR”**.

OUTPUT

The Serial Monitor displays the distance measured between the object in front of the sensor and the sensor itself. As the object moves further away from the sensor, this value increases. Moving the object closer to the sensor decreases the value read by the sensor.



DHT-22 Humidity and Temperature Sensor



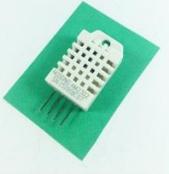
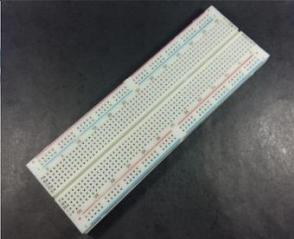
The DHT22 (also named as AM2302) is a basic low-cost digital temperature and humidity sensor. It uses a capacitive humidity sensor and a thermistor to measure the surrounding air, and spits out a digital signal on the data pin (no analog input pins needed). Connections are simple, the first pin on the left to 3-5V power, the second pin to your data input pin and the right most pin to ground.

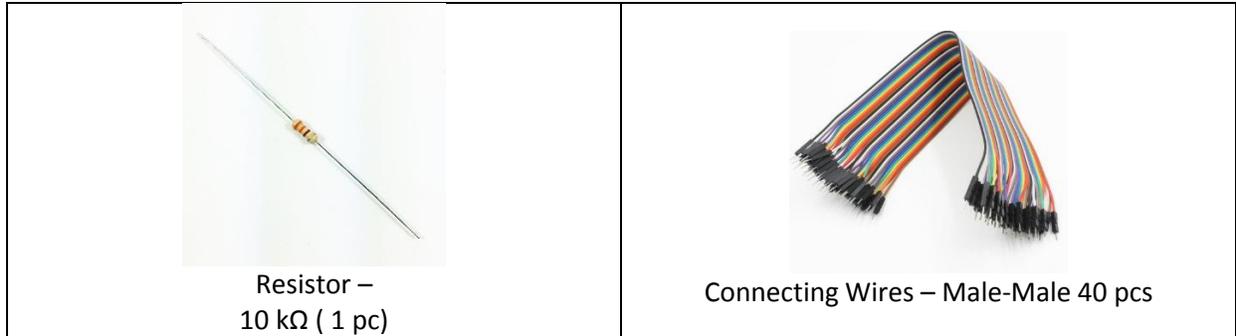
HARDWARE SPECIFICATIONS

- Power: 3-5V
- Max Current: 2.5mA
- Humidity: 0-100%, 2-5% accuracy
- Temperature: -40 to 80°C, $\pm 0.5^\circ\text{C}$ accuracy

PARTS LIST

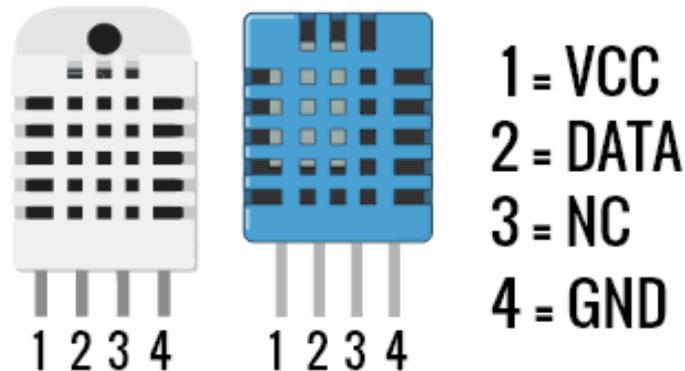
For this setup, we will need the following materials:

			
DHT-22 Temperature and Humidity Sensor	Arduino Uno R3	USB A to USB B Cable	Breadboard (Full-Size)



HARDWARE OVERVIEW

The DHT-22/DHT-11 Temperature Sensor have 3 pins to connect: VCC, DATA and GND.



The table below describes the function of each pin in the module

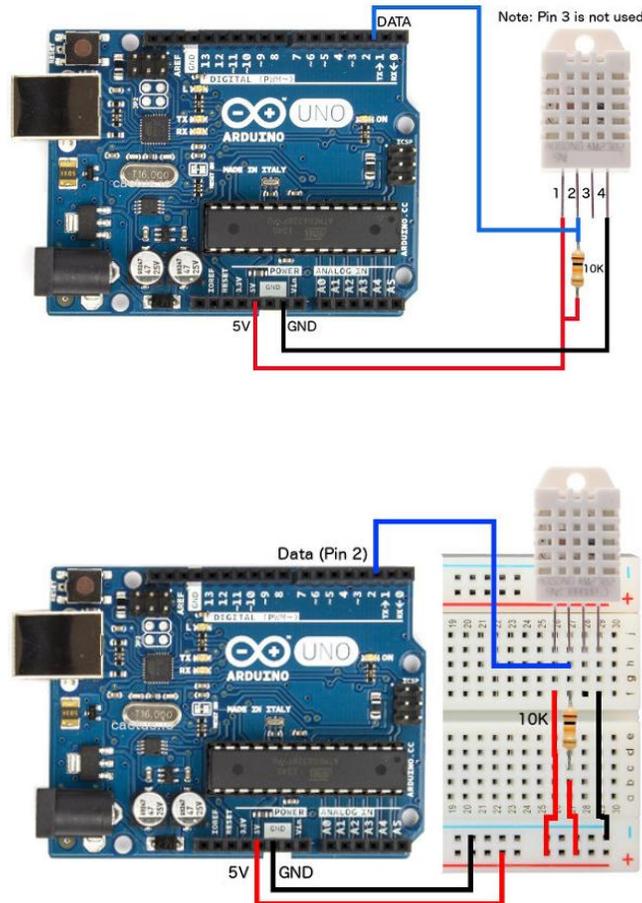
INPUT	Description
GND	To be connected to the GND pin in a microcontroller.
VCC	Supplies power to the module. Could be connected to a +5V pin.
OUTPUT	
DATA	Sends humidity and temperature data to microcontroller.

WIRING CONNECTION

<u>DHT22 Sensor</u>	---	<u>Arduino</u>
Pin 1	---	5V
Pin 2	---	DIGITAL PIN #2
Pin 3	---	No connection
Pin 4	---	GND (Ground)

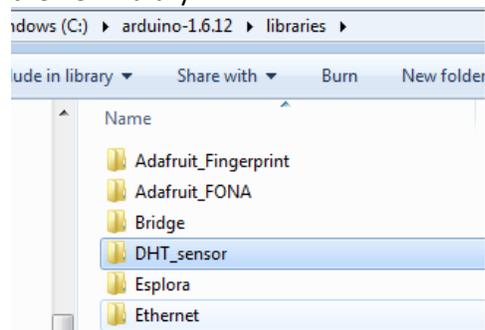
// Connect pin 1 (on the left) of the sensor to +5V
 // Connect pin 2 of the sensor to whatever your DHTPIN is
 // Connect pin 4 (on the right) of the sensor to GROUND

// Connect a 10K resistor from pin 2 (data) to pin 1 (power) of the sensor

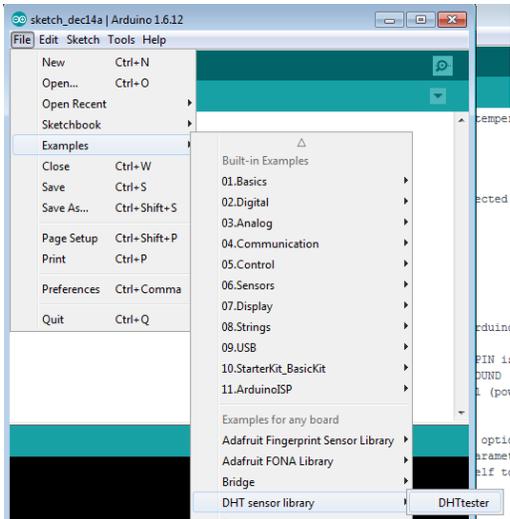


ARDUINO CODE

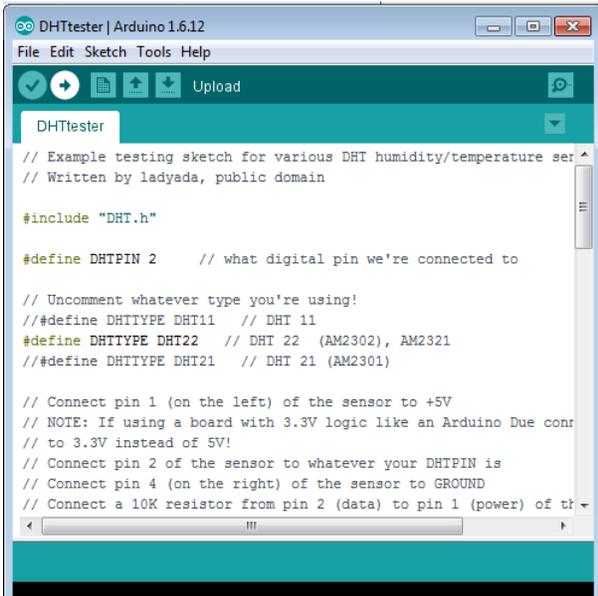
From the included Files, extract the folder from the zip file **DHT_sensor_library**. Copy the extracted file folder and paste it to your Arduino libraries folder (example: C:\arduino-1.6.5-r5\libraries). Rename the file folder to **DHT_sensor**. You also need to close and reopen all Arduino IDE before you can start using the new library.



Now open your Arduino IDE software from where you installed the DHT sensor library. Go to Files>Examples>DHT sensor library>DHTtester



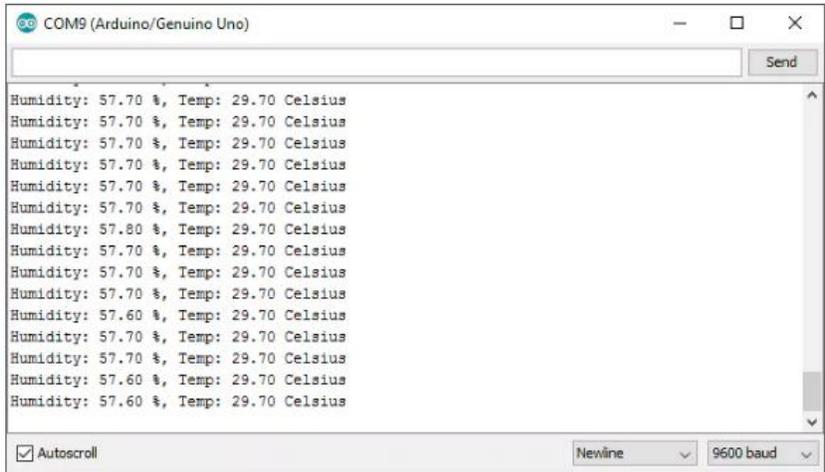
Compile and upload the Arduino code to your Arduino board.



OUTPUT

After uploading open the Arduino Serial Terminal and set the Baud speed to 9600.

You will see the output value with Temperature & Humidity every second of your current room temperature.



Flame/Fire/Light Sensor Module



Detect Flame/Fire/Light with this easy to use tiny board. You can easily connect it to popular controllers such as Arduino as an input digital pin giving a HIGH or LOW output signal. The board also has a mounting hole for mounting it to your desired surface.

HARDWARE SPECIFICATIONS

- Working voltage: 3.3-5V
- Detect flame/fire wavelength at 760nm ~ 1100nm
- Flame detect distance ~80cm
- Detect angle: 60 degrees
- Sensitivity adjust with onboard potentiometer trimmer
- Digital output 1 or 0 (from built in comparator)
- Mounting hole
- With comparator circuit based on LM393
- Dimension: Height: 7.3 mm, Width: 47.2 mm
- Weight 2.1 g

PARTS LIST

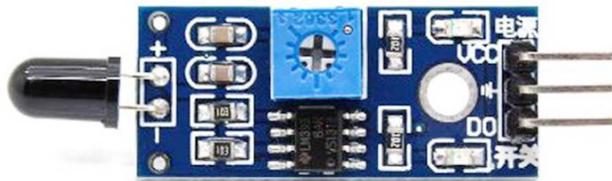
For this setup, we will need the following materials:



Flame/Fire/Light Sensor	Arduino Uno R3	USB A to USB B Cable
		
Connecting Wires – Male-Female 40 pcs		

HARDWARE OVERVIEW

The Flame/Fire/Light sensor has three pins: VCC, GND and DO



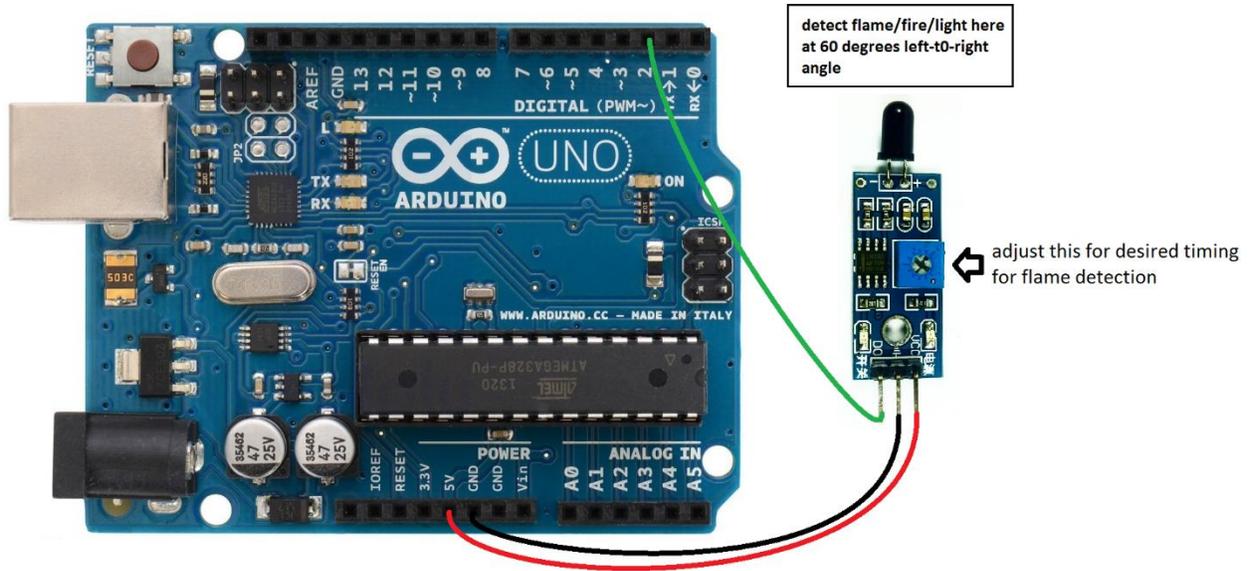
INPUT	Description
GND	To be connected to the GND pin in a microcontroller.
VCC	Supplies power to the module. Could be connected to a +5V pin or +3.3V pin.
OUTPUT	
DO	Stands for Digital Output. Gives a HIGH or LOW output signal when monitoring for fire/flame/light

WIRING CONNECTION

Connect the sensor to the Arduino Uno according to the table below:

<u>Flame/Fire/Light Sensor</u>	--->	<u>Arduino</u>
VCC	--->	5V
GND	--->	GND
DO	--->	DIGITAL PIN #2

FLAME/FIRE/LIGHT SENSOR MODULE



ARDUINO CODE

On the blank Arduino script, copy the following code:

```
int FlamePin = 2;
int Flame;
void setup() {
  // put your setup code here, to run once:
  pinMode(FlamePin, INPUT);
  Serial.begin(9600);
}

void loop() {
  // put your main code here, to run repeatedly:
  Flame = digitalRead(FlamePin);
  if (Flame == HIGH)
  {
    Serial.println("FIRE!");
  }
  else
  {
    Serial.println("NO FIRE");
  }
}
```

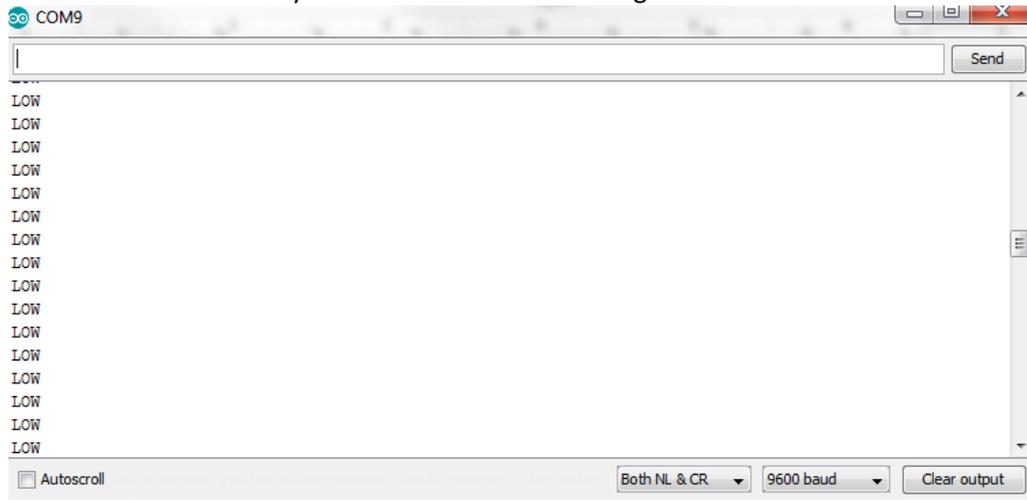
Upload the code. Once done, open Serial Monitor and set baud rate to **9600**.

OUTPUT

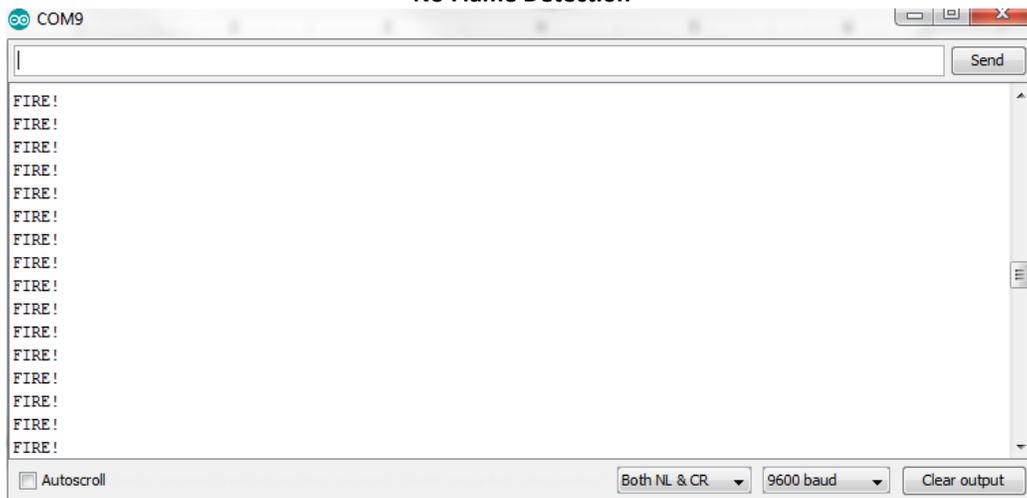
It will display if there is a flame/fire/light in front of the sensor. The sensor have about 60 degrees left to right detecting angle.

If you have the flame or light in front of the sensor you may need to adjust the detecting capability. The trimmer potentiometer (blue box with screw terminal figure) on the sensor board is the one you set for the detection. Try turning slowly the potentiometer from left to right until you have a detection and no detection for the flame or light that you are sensing.

You may try to put an output LED or output buzzer to light up or sound alarm if there is a flame detected. And remember to add your code for the LED blinking or buzzer alarm.

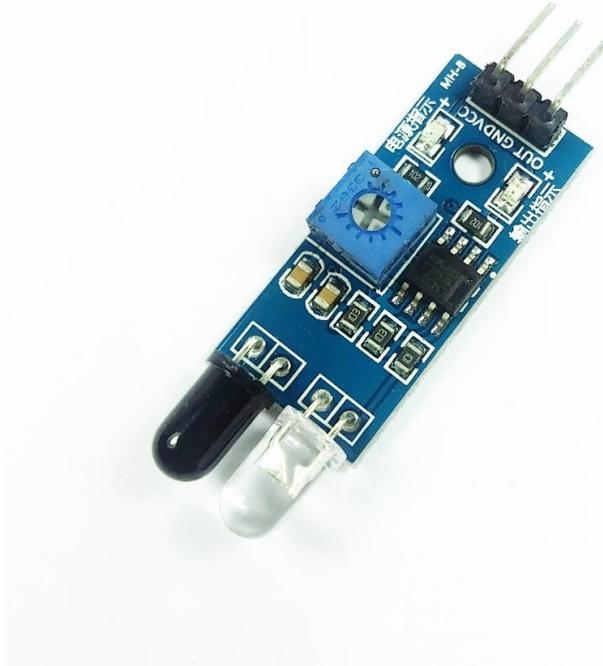


No Flame Detection



Flame Detected

Infrared Line Sensor Board (1 channel)



This Infrared Line or Obstacle Detection Sensor board is perfect for your line follower or obstacle detection robotic car projects. This is the 1 channel board version. When powered, you set detection level by adjusting the on-board trimmer potentiometer. You can get the Output data in digital HIGH and LOW when line is detected in the OUT (digital output) pin.

It has an on-board comparator circuit with adjustable potentiometer to set the reference/threshold voltage for the infrared detection level. It also has power and digital output LED indicator for easy debugging. The circuit is a pair of infrared emitter and phototransistor which send signal to each other when there is a white/clear surface and blocks any signal when there is black/dark surface.

In other applications, this sensor board can also be used for obstacle or wall detection projects.

HARDWARE SPECIFICATIONS

- Working Voltage: DC 3.3-5V
- Signal Output: 0 (detected), 1 (not detected)
- Detecting Distance: 2-30cm (adjustable)
- Detection Angle: degrees ~35
- Module Size: 32X14mm
- Use comparator LM393
- Board Size : 3.1CM * 1.5CM / 1.22 * 0.59"
- Dimension: Height: 7.3 mm, Width: 47.2 mm, Weight 2.1 g

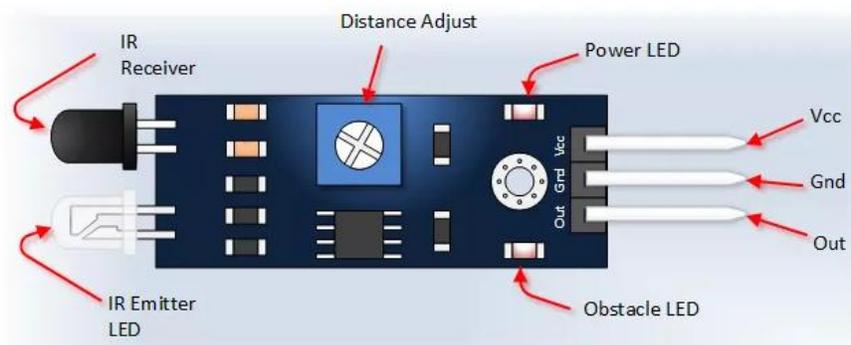
PARTS LIST

For this setup, we will need the following materials:

 <p>Line Follower Sensor Board 1 channel</p>	 <p>Arduino Uno R3</p>	 <p>USB A to USB B Cable</p>
 <p>Connecting Wires – Male-Female 40 pcs</p>		

HARDWARE OVERVIEW

The line follower sensor has three pins: VCC, GND and Out.



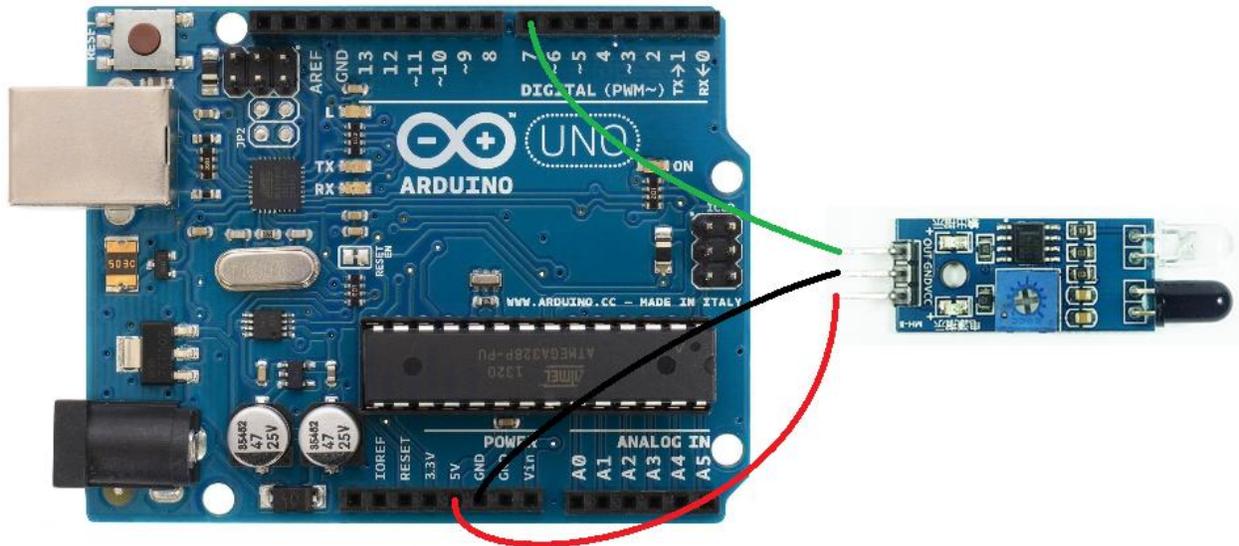
INPUT	Description
GND	To be connected to the GND pin in a microcontroller.
VCC	Supplies power to the module. Could be connected to a +5V pin or +3.3V pin.
OUTPUT	
Out	Sends a high or low pulse signal to indicate whether there is a black or a white color detected.

WIRING CONNECTION

Connect the line sensor to the Arduino Board according to the table below:

<u>Line/Obstacle Sensor</u>	---	<u>Arduino</u>
VCC	---	5V
GND	---	GND

OUT ---> DIGITAL PIN #7



ARDUINO CODE

This example will use the board as obstacle detection device.

This Line Sensor board can detect a black line in a white surface. This can be used for line follower robot projects. This same board can also be used for obstacle detection where you can detect a certain object or wall and do something after you detect the wall (example light and LED, sound a buzzer, or run an Arduino robotic car in different direction if you are building a robot).

For the following example code the use of the device is to detect a wall/obstacle and display a text alert. Copy the code below and paste it in your Arduino IDE. Compile and Upload the code to your Arduino board.

```
// IR Obstacle Collision Detection Module
// Henry's Bench
// Rewritten Example by BITSTOC

int LED = 13; // Use the onboard Uno LED
int isObstaclePin = 7; // This is our input pin
int isObstacle = HIGH; // HIGH MEANS NO OBSTACLE

void setup() {
  pinMode(LED, OUTPUT);
  pinMode(isObstaclePin, INPUT);
  Serial.begin(9600);
}

void loop() {
  isObstacle = digitalRead(isObstaclePin);
  if (isObstacle == LOW)
  {
    Serial.println("OBSTACLE Detected!!, OBSTACLE Detected!!");
    digitalWrite(LED, HIGH);
  }
}
```

```
else
{
  Serial.println("clear");
  digitalWrite(LED, LOW);
}
delay(200);
}
```

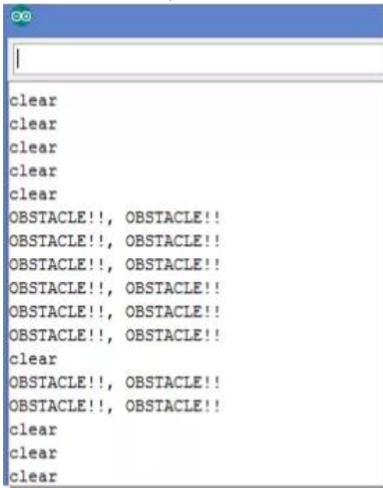
OUTPUT

After uploading the code open your Arduino Serial Terminal and set the Baud to 9600. It will display if it is clear or an obstacle is detected.

Now place your hand or object in front of sensor and it will display an "Obstacle Detected" message on the Serial Terminal. The LED on the sensor board and the LED on the Arduino board connected to PIN13 will also light up. When no obstacle detected it will display a "clear" message.

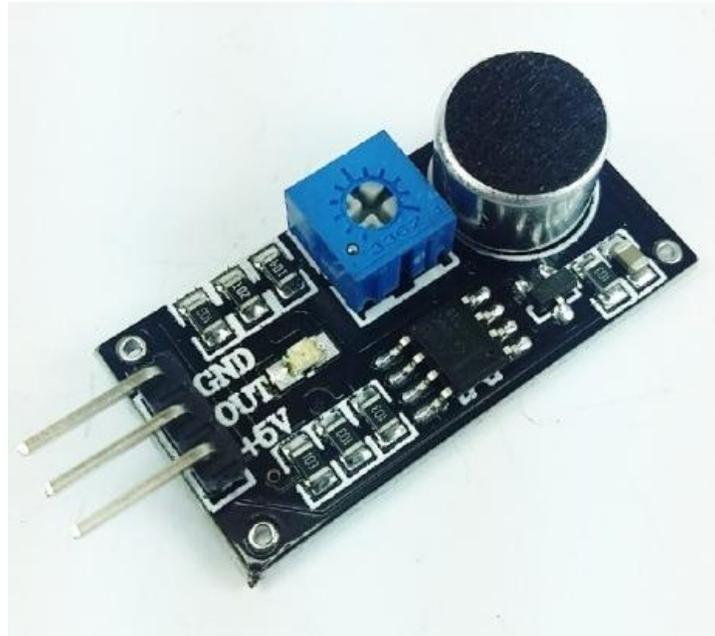
If you have your hand or the object to detect in front of the sensor you may need to adjust the detecting capability. The trimmer potentiometer (blue box with screw terminal figure) on the sensor board is the one you set for the detection level. Try turning slowly the potentiometer from left to right until you have a detection and no detection output on the Serial Terminal.

The result may look like below.



```
clear
clear
clear
clear
clear
OBSTACLE!!, OBSTACLE!!
OBSTACLE!!, OBSTACLE!!
OBSTACLE!!, OBSTACLE!!
OBSTACLE!!, OBSTACLE!!
OBSTACLE!!, OBSTACLE!!
OBSTACLE!!, OBSTACLE!!
clear
OBSTACLE!!, OBSTACLE!!
OBSTACLE!!, OBSTACLE!!
clear
clear
clear
```

Sound Sensor Module



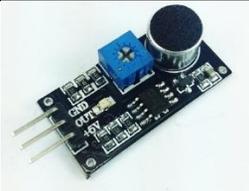
The Sound Sensor Module detects a sound nearby through its on-board electret microphone. It also has a comparator circuit which gives a HIGH (5V) or LOW (0V) signal. This only needs 3 pins. Supply Voltage (+5V), GND, and the digital OUTPUT which gives a HIGH or LOW signal when a sound is detected or not detected. The sound amplitude threshold/compare can be adjusted thru the on-board potentiometer so you can set the loudness to detect.

HARDWARE SPECIFICATIONS

- VCC = 5V
- 3pin DIGITAL Output
- With adjustable Sound amplitude to detect

PARTS LIST

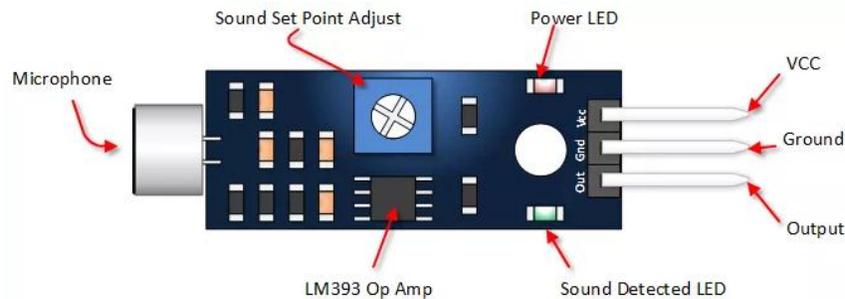
For this setup, we will need the following materials:

 <p>Sound Sensor</p>	 <p>Arduino Uno R3</p>	 <p>USB A to USB B Cable</p>
---	---	---



HARDWARE OVERVIEW

The line follower sensor has three pins: VCC, GND and Out.



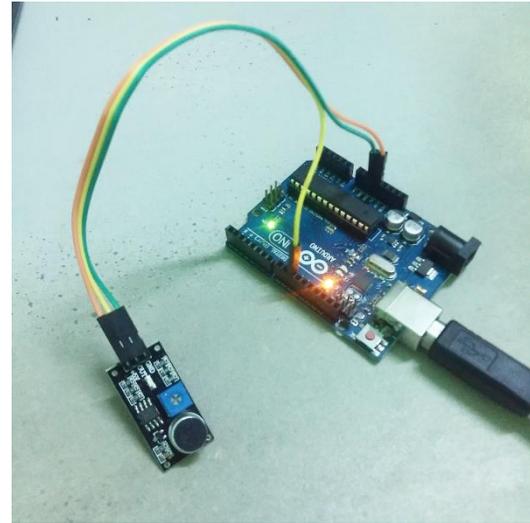
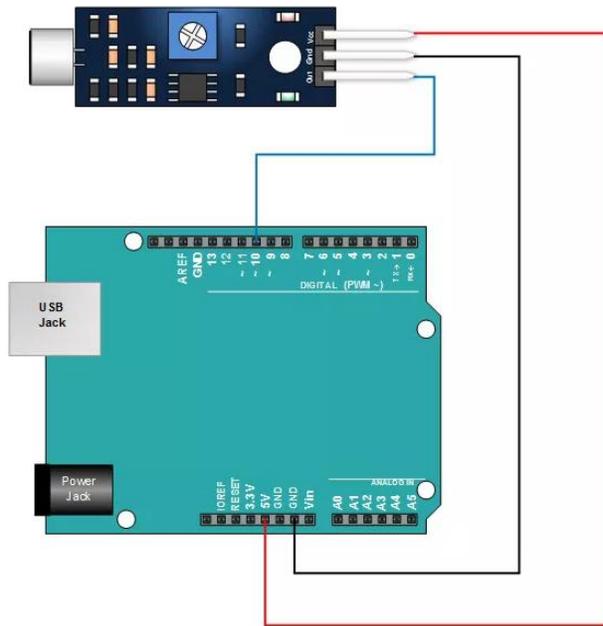
INPUT	Description
GND	To be connected to the GND pin in a microcontroller.
VCC	Supplies power to the module. Could be connected to a +5V pin or +3.3V pin.
OUTPUT	
Out	Sends a high or low pulse signal to indicate sound is detected or not.

To adjust sound sensitivity, rotating clockwise the potentiometer makes the sensor more sensitive, while rotating counterclockwise makes the sensor less sensitive.

WIRING CONNECTION

Connect the line sensor to the Arduino Board according to the table below:

<u>Sound Sensor Module</u>	---	<u>Arduino</u>
VCC	---	5V
GND	---	GND
OUT	---	DIGITAL PIN #10



ARDUINO CODE

Copy the code below into and paste it in a new sketch of the Arduino IDE.

```
//Henry's Bench
//Arduino Sound Detection Sensor Module
//Modified and Rewritten Example by BITSTOC

//Declare variables to use
int soundDetectPin = 10; // Use Pin 10 as our Input
int soundDetectState = HIGH; // Use this variable to store the sound state if HIGH or LOW
boolean bAlarm = false;

//We will use this variable to compare from sound detected to no sound detected
unsigned long lastSoundDetectTime; // Record the time when we detected a sound

int soundAlarmTime = 500; // Number of milli seconds to keep the sound alarm high

void setup ()
{
  //Set a Serial Monitor Baud Rate to monitor the output. Declare soundDetectPin sensor pin as Input
  Serial.begin(9600);
  pinMode(soundDetectPin, INPUT);
}

void loop ()
{
  soundDetectState = digitalRead(soundDetectPin) ; // read if there is a sound detected

  //We use command if/else statement to compare if sound or no sound detected
  //When there is a sound detected by the sound sensor, the sound sensor OUT pin will go LOW ( GND)
  if (soundDetectState == LOW) // If loud sound is detected
  {
    lastSoundDetectTime = millis(); // record the time when we hear the sound in milliseconds
    // The boolean bAlarm is used to compare a state if there is a sound or no sound
  }
}
```

```

// and show a text in the Serial Monitor corresponding to the state
if (!bAlarm) {
  Serial.println("LOUD, LOUD");
  bAlarm = true;
}
}
else
{
  if ( (millis() - lastSoundDetectTime) > soundAlarmTime && bAlarm) {
    Serial.println("quiet");
    bAlarm = false;
  }
}
}
}
}

```

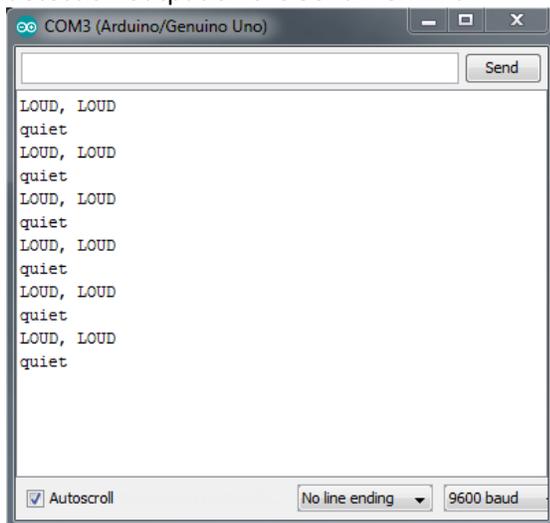
Compile the code, then upload it to the Arduino Uno board.

OUTPUT

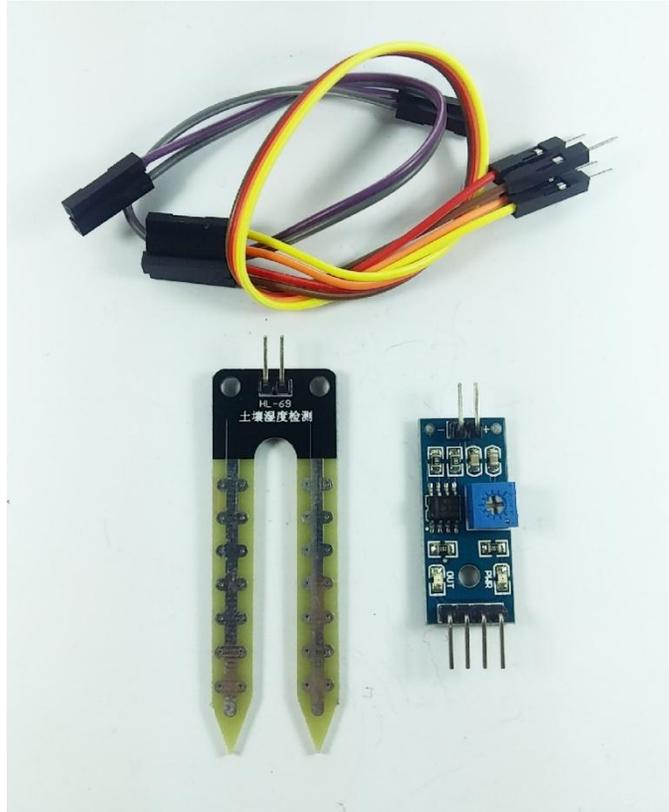
After uploading the code open your Arduino Serial Terminal and set the Baud to 9600. It will display a text if the sensor module detect a LOUD sound or if it's "quite" or not detecting any sound.

Now place your hand near the sound sensor and try clapping. If the sound you produced from clapping is loud enough as loud detection for the sensor it will display in the monitor a "LOUD, LOUD" text.

If you have your hand or any sound output you want to detect (like music or environmental noise) near the sensor and is not detected you may need to adjust the detecting capability. The trimmer potentiometer (blue box with screw terminal figure) on the sensor board is the one you set for the detection level. Try turning slowly the potentiometer from left or right until you have a detection and no detection output on the Serial Terminal.



Soil Moisture Sensor Kit



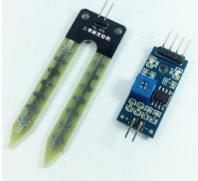
The soil moisture sensor or the hygrometer is usually used to detect the humidity or wetness of the soil. This sensor is perfect for building an automatic watering system or to monitor the soil moisture of your plants. The sensor kit is set up by two pieces: the electronic board, and the probe with two pads, that detects the water content. The sensor has a built-in potentiometer for sensitivity adjustment of the digital output (D0), a power LED and a digital output LED to use for viewing the output state of the sensor board.

HARDWARE SPECIFICATIONS

- Operating voltage: 3.3 V ~ 5 V
- Dual output mode, analog output or digital output
- Fixed bolt hole for easy installation
- Power indicator (red) and digital output indicator (green)
- LM393 Comparator based chip
- PCB panels Size: about 1.5 cm; Soil probe Dimensions: approx. 6 cm x 3 cm

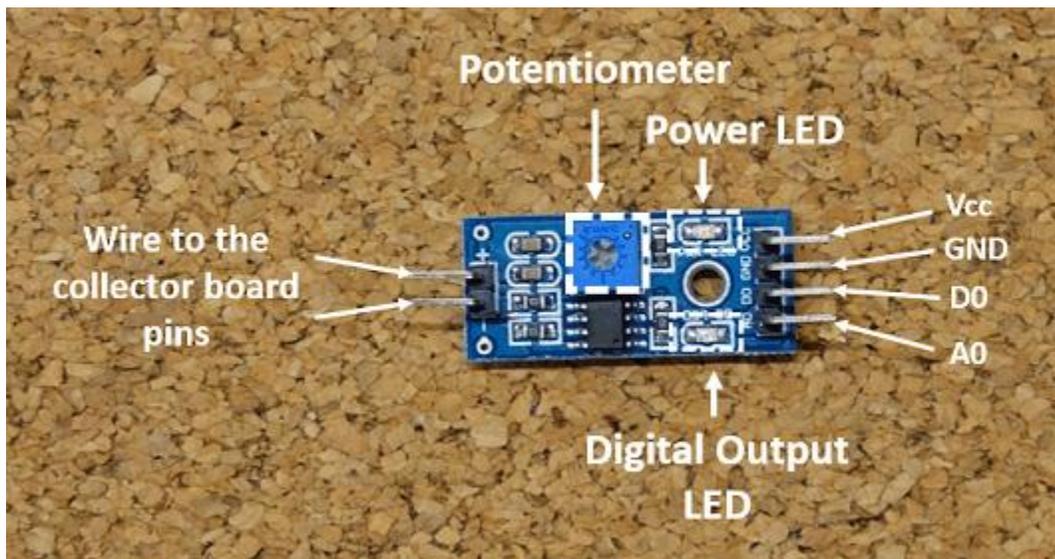
PARTS LIST

For this setup, we will need the following materials:

 <p>Soil Moisture Sensor</p>	 <p>Arduino Uno R3</p>	 <p>USB A to USB B Cable</p>	 <p>Breadboard (Full-Size)</p>
 <p>Resistor – 330 Ω (2 pc)</p>	 <p>Connecting Wires – Male-Male 40 pcs</p>	 <p>Connecting Wires – Male-Female 40 pcs</p>	 <p>Light Emitting Diode – Red (1 pc) Green (1 pc)</p>

HARDWARE OVERVIEW

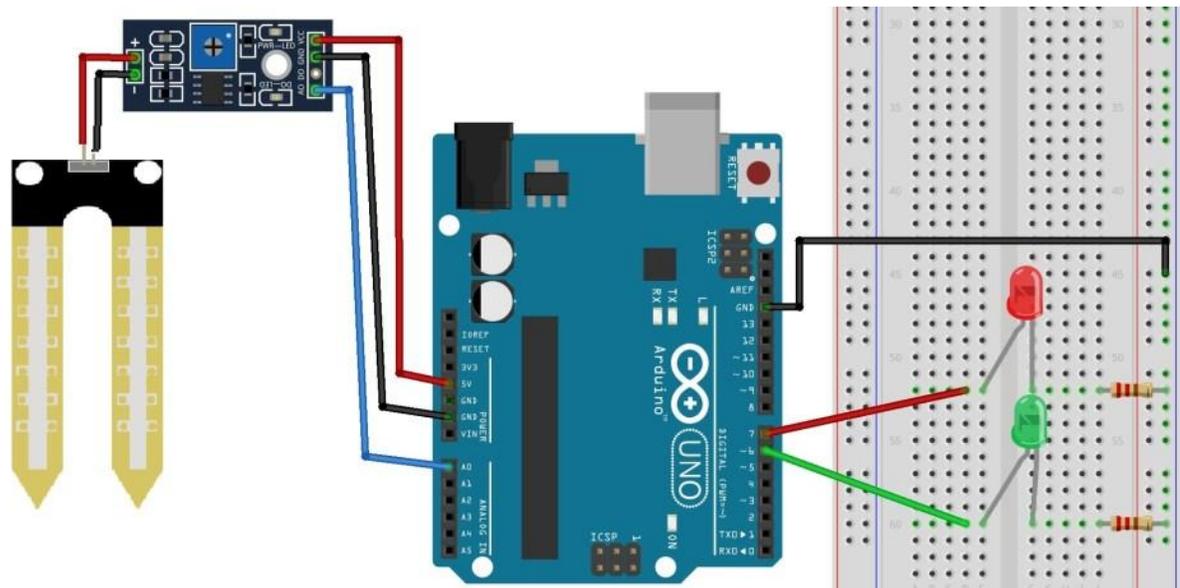
The Soil Moisture Board has 4 pins to connect to a microcontroller: VCC, GND, DO and AO.



PREPARING THE ARDUINO IDE

Connect the line sensor to the Arduino Board according to the table below:

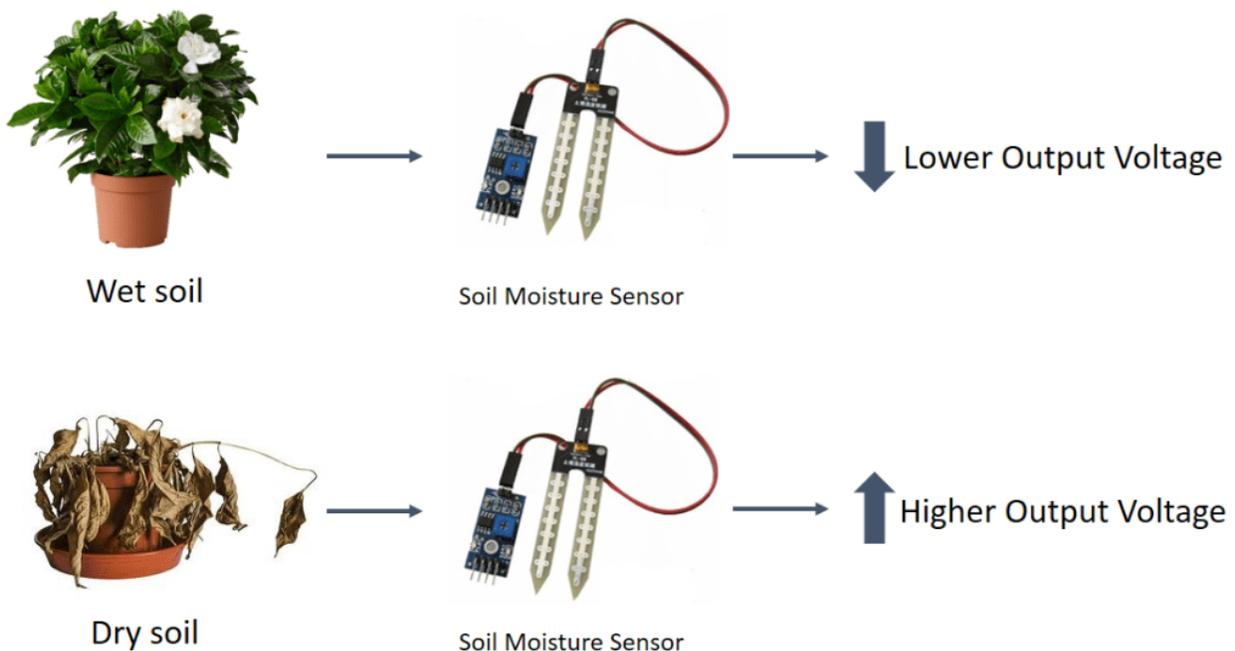
<u>Soil Moisture Sensor</u>	---	<u>Arduino</u>
VCC	---	5V
GND	---	GND
AO (output analog)	---	DIGITAL PIN #10
DO (output digital)	---	(optional use, any Digital pin)



The operation of the soil moisture sensor is as follows:

Wet: the output voltage decreases

Dry: the output voltage increases



The soil moisture sensor kit will read the wetness or dryness of the soil (or any setup that have wet/dry state) through the two-prong probe. If the probe detects a wet soil it will send a Low Output Voltage (Analog Voltage) to the Arduino. If the prove detects a dry soil it will send a High Output Voltage to the Arduino. Then from that voltage output you can do something like light up an LED if the soil is wet or dry.

This sensor is a perfect module to use for automated watering system for your garden or plants or even just display the condition on the soil of the plants or land.

ARDUINO CODE

Copy the code below into and paste it in a new sketch of the Arduino IDE.

```

/*
  By: randomnerdtutorials.com
  Rewritten and Edited for Example by BITSTOC
*/

//Declare variables to use
int soilsensorPin = A0; // this is the soil moisture sensor pin
int greenLED = 6; // green led to know for wet soil
int redLED = 7; // red led to know for dry soil

// adjust this value to desired level of wetness to detect (may require test before/during the
actual project)
int thresholdValue = 800;

// setup the input, output pins and the serial output text display monitoring in the computer
void setup() {
  pinMode(soilsensorPin, INPUT);
  pinMode(greenLED, OUTPUT);
  pinMode(redLED, OUTPUT);
  digitalWrite(greenLED, LOW);
  digitalWrite(redLED, LOW);
  Serial.begin(9600);
}

void loop() {
  // read the input on analog pin 0 (Arduino Analog pin A0)
  int sensorValue = analogRead(soilsensorPin);

  //print the value read by the sensor
  Serial.print(sensorValue);

  //compare the output using if/else statement to know if the soil is wet or dry
  //with designated text output on the monitor
  if (sensorValue < thresholdValue) {
    Serial.println(" - Doesn't need watering");
    digitalWrite(redLED, LOW);
    digitalWrite(greenLED, HIGH);
  }
  else {
    Serial.println(" - Time to water your plant");
    digitalWrite(redLED, HIGH);
    digitalWrite(greenLED, LOW);
  }
  delay(500); // give a little delay before doing the next sensor reading.
}

```

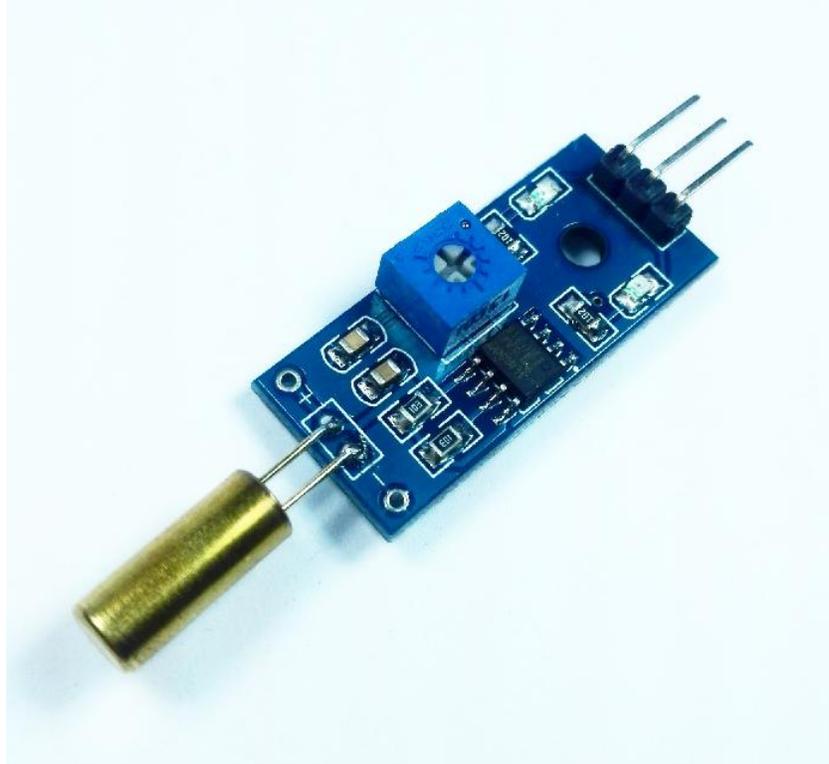
Compile the code, then upload it to the Arduino Uno board.

OUTPUT

After uploading the code open your Arduino Serial Terminal and set the Baud to 9600. The Arduino will start running the code and read the wetness or dryness of the soil. It will then display a text in the Arduino Serial Monitor if the plant needs watering or not.

Note that you may adjust the threshold or detection level of the soil moisture sensor as to how much voltage corresponds to wet or dry state. You can do this by changing the value of ***int thresholdValue = 800;*** in your code where you can set it from 0 to 1023. Where 0 is wet and 1023 is dry.
That's it! You just made an automation sensor device to take care of your plants and garden!

Tilt Sensor Module – FZ0722



This Tilt Sensor module can sense tilt angle or movement (changes in the angle of the object). It is roller-ball type tilt sensor consists of two conductive elements (poles) and a conductive free mass (rolling ball) that opens or closes connection with circuit.

The PCB has the size of 3.2 cm * 1.4cm. This module use LM393 comparator IC. The blue small box is the sensitivity potentiometer. It has LED indicator, the Power LED and the Switch signal LED. The headers are for VCC, GND and Digital Output from the on-board comparator. This board also has holes for easy installation.

HARDWARE SPECIFICATIONS

- Sensor can sense changes in body angle
- Adjustable sensitivity (shown in blue digital potentiometer adjustment)
- Operating Voltage 3.3V-5V
- Output format: digital switching outputs (0 and 1)

PARTS LIST

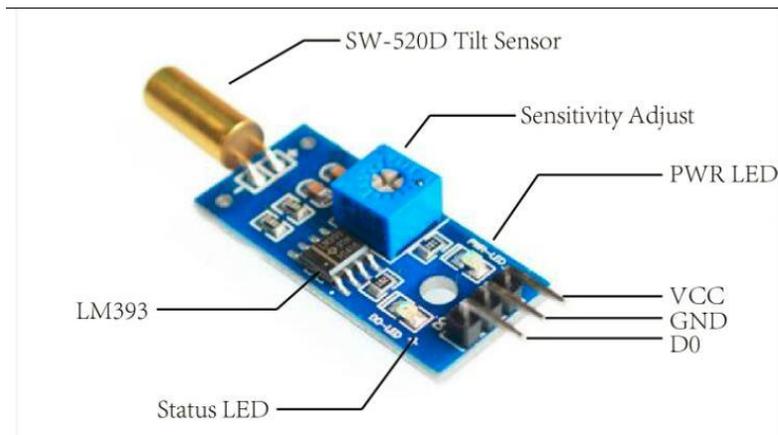
For this setup, we will need the following materials:

 <p>Tilt Sensor</p>	 <p>Arduino Uno R3</p>	 <p>USB A to USB B Cable</p>
--	---	---



HARDWARE OVERVIEW

The line follower sensor has three pins: VCC, GND and Out.

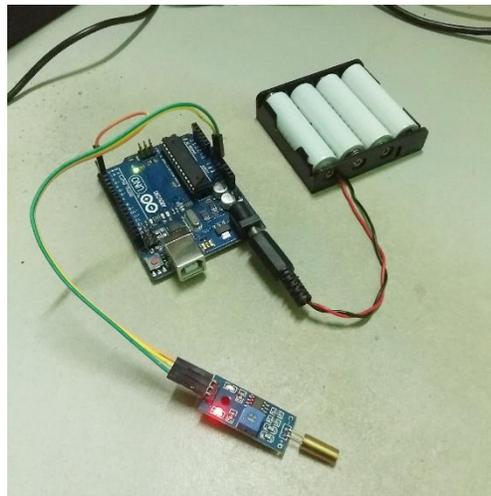
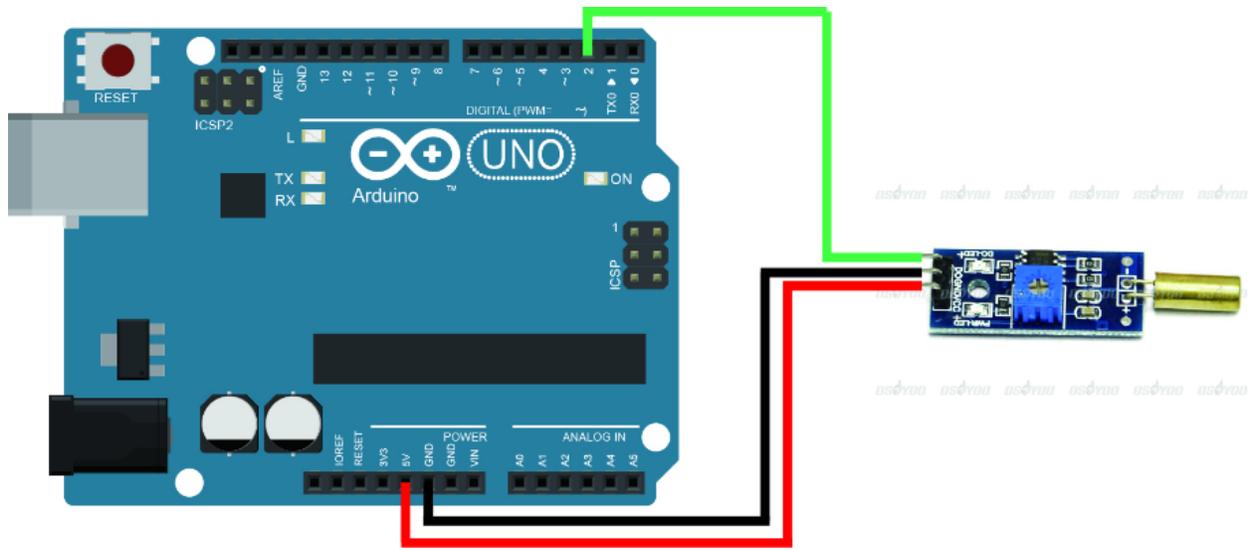


INPUT	Description
GND	To be connected to the GND pin in a microcontroller.
VCC	Supplies power to the module. Could be connected to a +5V pin or +3.3V pin.
OUTPUT	
DO	Sends a high or low pulse signal to indicate whether there the sensor is tilted or not

WIRING CONNECTION

Connect the line sensor to the Arduino Board according to the table below:

<u>Soil Moisture Sensor</u>	---	<u>Arduino</u>
VCC	---	5V
GND	---	GND
DO (output digital)	---	DIGITAL PIN #2

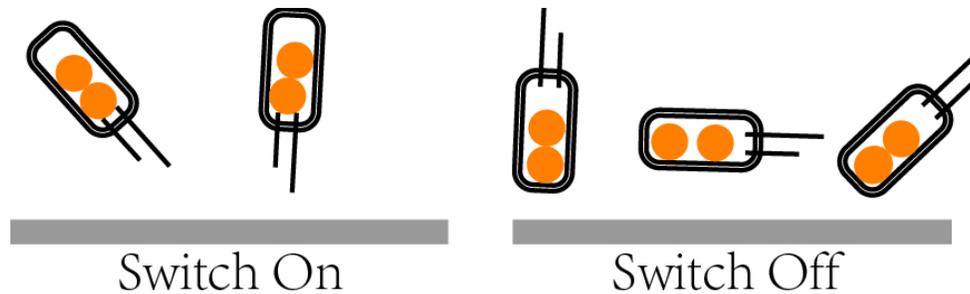


When the tilt sensor is:

Horizontal (with the ground): Steady State, No tilt, D0 pin in the sensor board is HIGH (5V), LED on the sensor board is OFF.

Vertical (turned pointing upwards): Tilt state changed, Tilt angle change detected, D0 pin in the sensor will turn LOW (0 V), LED on the sensor board will turn ON.

The tilt sensor module will detect an angle change (tilt) from horizontal to vertical position and vice versa. When the board is horizontally oriented the board will output HIGH on the D0 output pin and the onboard LED is OFF. When the board is turned or pointed upwards (angle changed) it will output a LOW state on the D0 output pin to feed to the Arduino and the onboard LED will turn ON to show that there is an angle change detected.



The tilt (angle) sensor is a perfect module to use when you want to detect a platform that is turned upside down, or detect a door fully open or fully closed and many more.

ARDUINO CODE

Copy the code below into and paste it in a new sketch of the Arduino IDE.

For the following example code the use of the device is to detect a wall/obstacle and display a text alert. Copy the code below and paste it in your Arduino IDE. Compile and Upload the code to your Arduino board.

```

/*
  From osoyoo.com
  Rewritten and Edited for Example by BITSTOC January 2018
*/

//Assign variables to use. We try using "const" keyword, to make sure a pin assigned will not change
//See "const" explanation here: https://www.arduino.cc/reference/en/language/variables/variable-scope--qualifiers/const/
const int sigPin = 2;    // Arduino digital input pin to read from the tilt sensor output
const int ledPin = 13;   // LED on the Arduino board to light on/off if sensor is tilted or not

// variable that will change
boolean sigState = 0; // variable for reading the tilt switch status. tilted or not tilted (0 or 1) (HIGH or LOW)

void setup()
{
  // assign frequency baud rate to view the output result in with text.
  Serial.begin(9600);
  // initialize the LED pin as an output:
  pinMode(ledPin, OUTPUT);
  // initialize the tilt switch pin as an input:
  pinMode(sigPin, INPUT);
}

void loop()
{
  // read the state of the tilt switch value:
  sigState = digitalRead(sigPin);
  if (sigState == HIGH)
  {
    // turn LED on:
    digitalWrite(ledPin, LOW);
    Serial.println("Tilt not detected, no angle angle changed");
  }
  else
  {
    // turn LED off:
    digitalWrite(ledPin, HIGH);
    Serial.println("Tilt detected, angle changed!");
  }
}

```

```
}

```

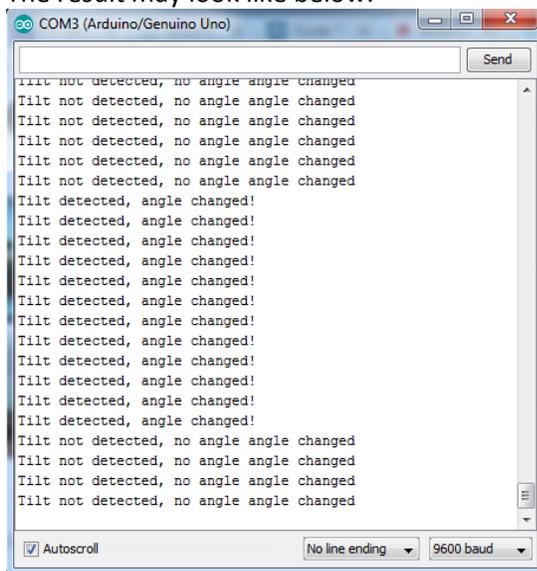
Compile the code, then upload it to the Arduino Uno board.

OUTPUT

After uploading the code open your Arduino Serial Terminal and set the Baud to 9600. The Arduino will start running the code and read the tilt sensor if there is an angle change or not. It will then display a text in the Arduino Serial Monitor to its corresponding state/position.

You may adjust the threshold or detection level of the angle change (tilt detection) for your specific application. The trimmer potentiometer (blue box with screw terminal figure) on the sensor board is the where you can set this up. Try turning slowly the potentiometer from left or right until you have a detection and no detection state that you desire.

The result may look like below.



```

COM3 (Arduino/Genuino Uno)
Tilt not detected, no angle angle changed
Tilt detected, angle changed!
Tilt not detected, no angle angle changed
  
```

Relay Module Opto-Isolated – 2 Channel



The 2-channel relay module is best use for your home automation and switching projects with high currents and load. Use this for turning on and off some high-powered devices and appliances thru and Arduino or any microcontroller. The Input of this relay module is logic LOW (drive a GND/LOW/ Zero Volts) in the Input pins to activate the relay to Open and Close circuit.

HARDWARE SPECIFICATIONS

- VCC = 3.3 or 5V
- Logic Low Activation
- Load capacity of 10A at 125-250VAC
- Load capacity of 10A at up to 30VDC
- with LED light indicator if activated or not activated

PARTS LIST

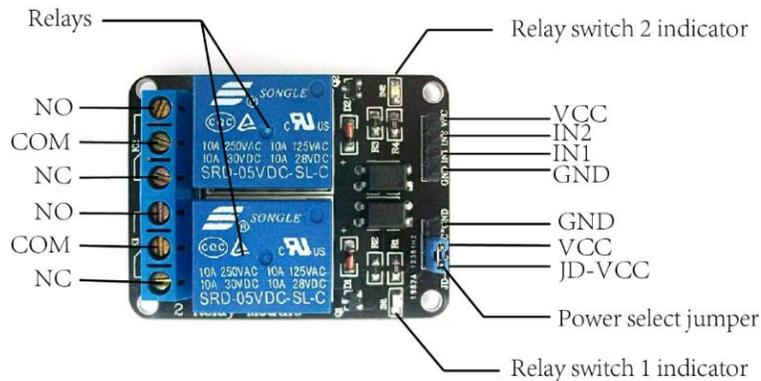
For this setup, we will need the following materials:

- 1 – Arduino Uno: <https://www.bitstoc.com/product/1/>
- Connecting wires : <https://www.bitstoc.com/product/107/>
- 1 – 2 channel relay module
- 1 – AC light bulb
- 1 – Bulb socket with AC plug

HARDWARE OVERVIEW

The 1 channel relay have 3 pins to be connected to the microcontroller: VCC, GND and IN. For the system to be controlled, the pins NO, COM and NC are used.

RELAY MODULE OPTO-ISOLATED – 2 CHANNEL



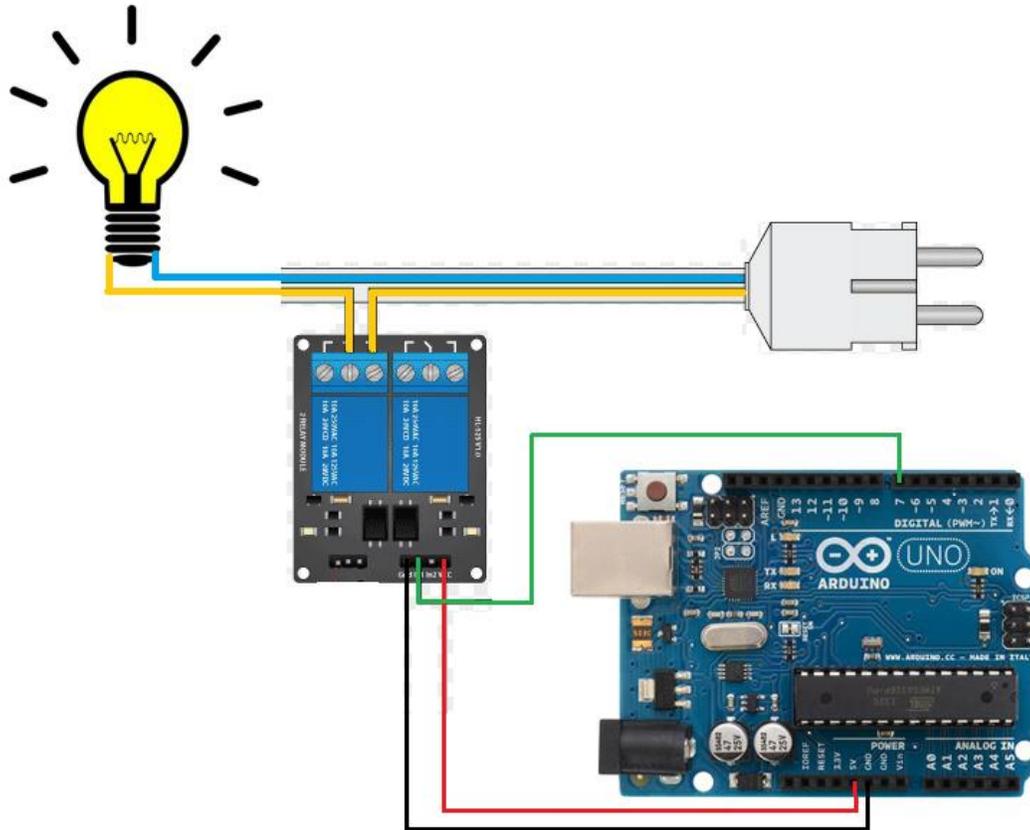
The table below describes the function of each pin in the module

INPUT	Description
IN1, IN2	Controls the state of the relay. This is connected to a digital pin in Arduino Uno or GPIO in Raspberry Pi.
GND	To be connected to the GND pin in a microcontroller.
VCC	Supplies power to the module. Could be connected to a +5V pin or +3.3V pin.
OUTPUT	
NO	Normally Open pin in a relay. When relay is in off state, this pin stays open with respect to COM pin. When relay is in on state, this pin connects with COM pin.
COM	Common pin. Connected to one part of the system to be controlled.
NC	Normally Closed pin in a relay. When relay is in off state, this pin stays close with respect to COM pin. When relay is in on state, this pin disconnects with COM pin.

WIRING CONNECTION

Setup the circuit as shown below:

Relay Board	--->	Arduino
VCC	--->	5V
GND	--->	GND
IN1	--->	DIGITAL PIN# 7



Other connection

Small bulb socket with 2 wires. 1 of the 2 socket wire is open stripped and cut. One cut end is inserted and screwed to COM “common” port of the relay (middle of the 3pin blue colored plastic screw terminal). The other cut end is inserted and screwed to NO “Normally-Open” port of the relay board (left side of the COM pin facing the 3pin screw terminal). Note that this project may involve high voltage so precautionary measures are needed.

WARNING



HIGH VOLTAGE

Incorrect or improper use could result in:

- serious injuries or death
- physical damage of the product
- creating dangerous hazards



How it works:

Relays can be used for controlling (turn on and off) a high power device such as electric fan, TV monitor, bulbs and other high power devices whether AC or DC. The Arduino can only handle small current so we use a relay module to receive the high/low small signal from the Arduino and the relay will be the one to drive the high voltage and current of the load device.

Putting "LOW" (0V) to the input (IN1, IN2) of the relay, will activate the relay. It will close the Normally Open (NO) port and it will open the Normally Close (NC) port of the relay. When the relay is activated a corresponding indicator LED will also light up on the relay channel that is activated.

ARDUINO CODE

Open Arduino IDE. Set the board to Arduino/Genuino Uno. Copy the code below to the sketch file:

```

/*
  By howtomechatronics
  Rewritten and Edited for Example by BITSTOC
  Arduino and Relay Arduino Code
*/

//declare variable to use
int relayIN1 = 7; // use Arduino Digital PIN 7 and set as variable
"relayIN1"

```

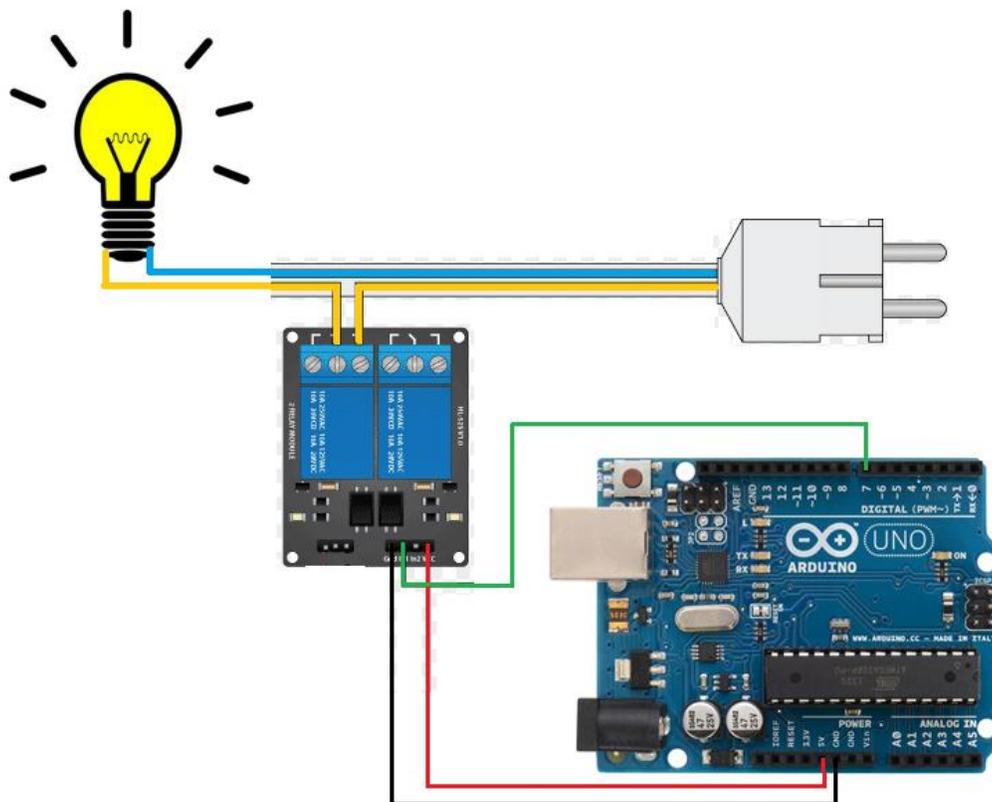
```

void setup() {
  pinMode(relayIN1, OUTPUT); // declare arduino pin 7 as an output port
  digitalWrite(relayIN1, HIGH); // set HIGH initially for relay as not
  activated
}

void loop() {
  digitalWrite(relayIN1, LOW); // drive LOW to activate the relay
  delay(3000); // wait for 3 seconds
  digitalWrite(relayIN1, HIGH); // drive HIGH to deactivate the relay
  delay(3000); // wait another 3 seconds
}

```

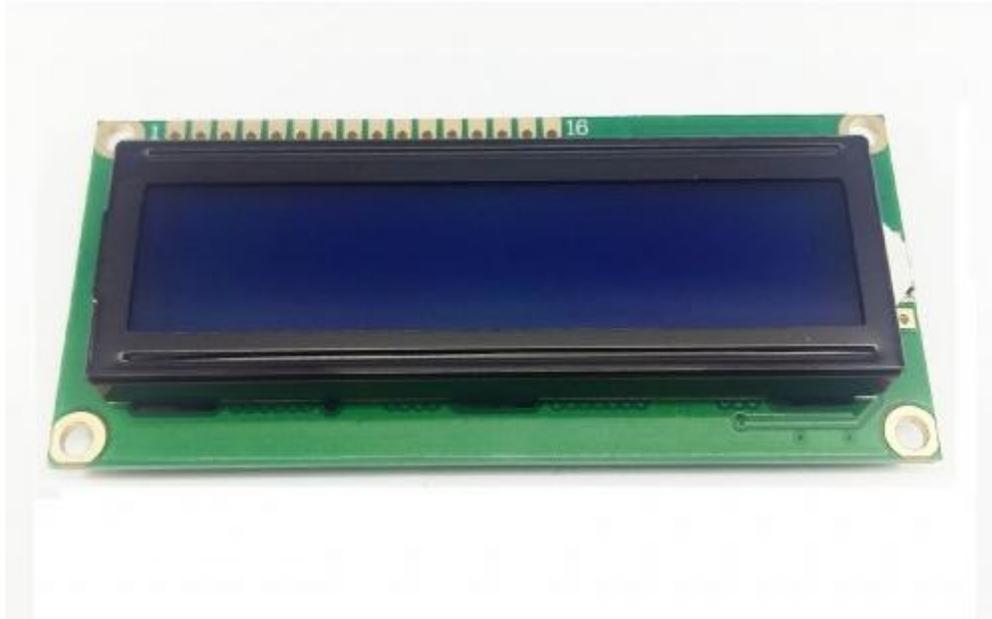
OUTPUT



When the code starts running the bulb should light ON and light OFF every 3 seconds.

The Arduino Digital PIN 7 will output a logic LOW to the IN1 pin of the Relay module board, thus activating the channel 1 relay on the board. The corresponding LED indicator for the relay channel will also light up. When relay channel 1 is activated, the **NO (Normally Open)** pin and the **COM** pin on the relay load side (blue screw terminal) will connect to each through the inside of the relay. Since the sliced wire is now electrically connected there will now be a continuous electricity to flow from the wall outlet to the bulb.

LCD 2x16 Character Module



The LCD Module is a display module for displaying data, be it in the form of numbers, texts or symbols. You can display 16 characters in a single row, with two rows available.

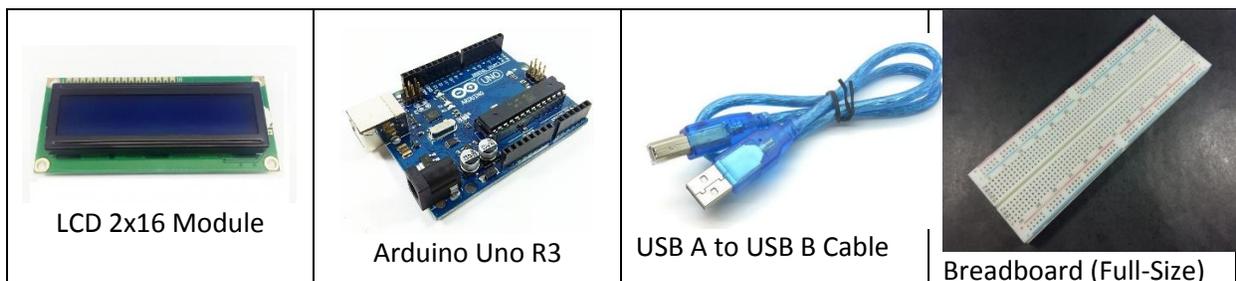
It is also easily integrated with popular boards such as Arduino as libraries are readily available for your use. Use this to display data anywhere, even in dark places as it has a backlight LED.

HARDWARE SPECIFICATIONS

- Working Voltage: DC 4.7-5V
- Blue Backlight
- Current consumption without backlight: 1mA
- Alpha numeric display module, displays numbers and alphabet in 5x8 pixel box

PARTS LIST

For this setup, we will need the following materials:



 <p>Resistor – 330 Ω (1 pc)</p>	 <p>Connecting Wires – Male-Male 40 pcs</p>	 <p>Potentiometer – 10 kΩ (1 pc)</p>
---	--	---

HARDWARE OVERVIEW



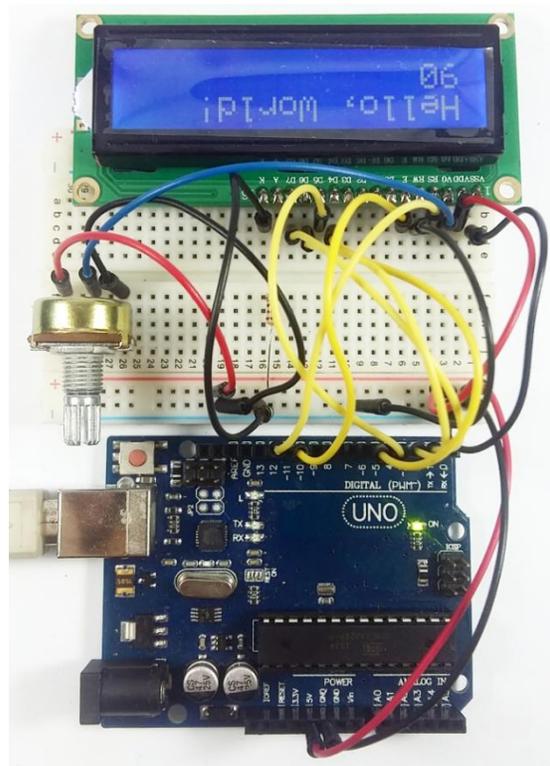
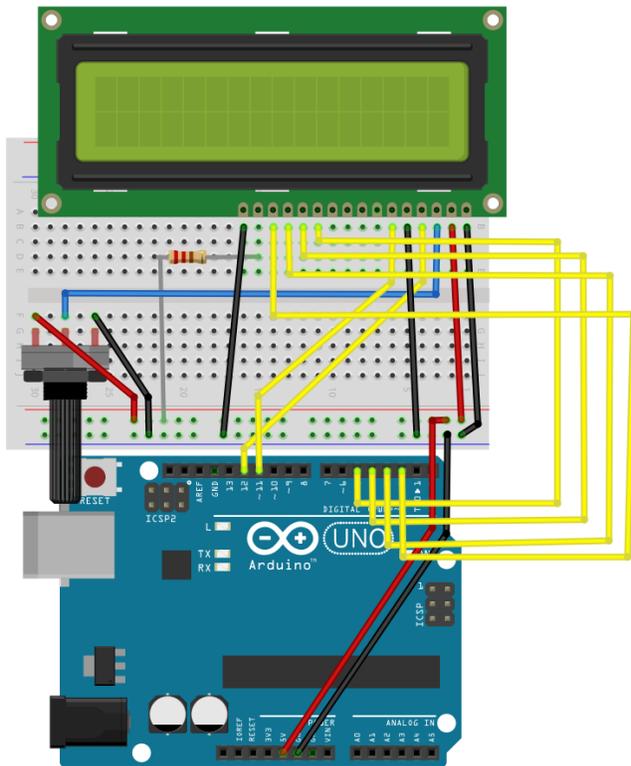
INPUT	Description
VDD	Connects to supply, should be at 5V.
VSS	Connects to GND.
DATA PIN (0-8)	Data pins 0-7 form an 8-bit data line. Can also operate as a 4-bit data line by connecting 0-3, leaving 4-7 free.
LED Positive	Backlight LED positive pin. Equivalent to anode of an LED.
LED Negative	Backlight LED negative pin. Equivalent to cathode of an LED.

WIRING CONNECTION

Connect the LCD Display module to the Arduino Board according to the table below:

- LCD pin 1 (VSS) to GND
- LCD pin 2 (VDD) to 5V+
- LCD pin 3 (V0) to Center of potentiometer
- LCD pin 4 (RS) to Arduino pin 12
- LCD pin 5 (RW) to GND

LCD pin 6 (E/Enable) to Arduino pin 11
 LCD pin 7-10 (D0-D3) no connection
 LCD pin 11 (D4) to Arduino pin 5
 LCD pin 12 (D5) to Arduino pin 4
 LCD pin 13 (D6) to Arduino pin 3
 LCD pin 14 (D7) to Arduino pin 2
 LCD pin 15 (A,Anode) to 300 ohm resistor to 5V+
 LCD pin 16 (K,Cathode) to GND
 *one side of potentiometer to 5V+ and one side to GND



ARDUINO CODE

The example code below will display a “Hello, World!” message on the LCD module. To do this, open Arduino. On the sketch file, copy the code below:

```

/*
  Project 8 - LCD Display
*/

// We use the "#include" command to use an LCD library "LiquidCrystal.h"
// Always include this line when using an LCD display
#include <LiquidCrystal.h>

// An LCD needs some initialization pins, from the Arduino board
// we use the pins listed in the "lcd()" function below
LiquidCrystal lcd(12, 11, 5, 4, 3, 2);

void setup() {
  // To start using the LCD, we first need to set up the LCD's number of columns and rows:
  // thus, put (16 columns and 2 rows) using the "lcd.begin(x,y)" command
  lcd.begin(16, 2);
}

```

```
// Now, we can print a message to the LCD.  
lcd.print("Hello, World!");  
}  
  
void loop() {  
  // set the cursor to column 0, line 1  
  // (note: line 1 is the second row, since counting begins with 0):  
  lcd.setCursor(0, 1);  
  // We test out by printing the number of seconds since the time we start or reset the Arduino  
  lcd.print(millis() / 1000);  
}
```

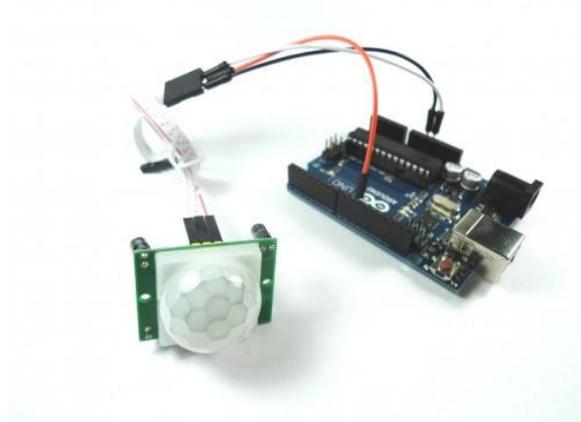
Once done, check the LCD module.

OUTPUT

On the LCD module, the first row displays a text message **“Hello, World!”** The second line displays the time that passed since the code started running. This code resets every time the Arduino board resets. This is in terms of seconds.



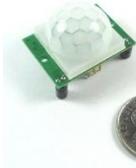
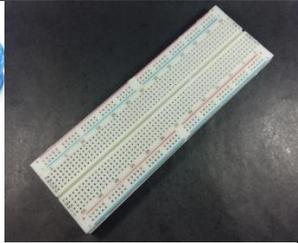
PROJECT 1: MOTION ACTIVATED ALARM SYSTEM



One of the most common systems using PIR technology is the motion activated alarm system. This system makes use of PIR for detecting movement in its detection range. If there is any movement detected, alarms in the form of lights or sound go off. This system is used as an anti-burglar system, or as a subsystem for the security of a building.

PARTS LIST

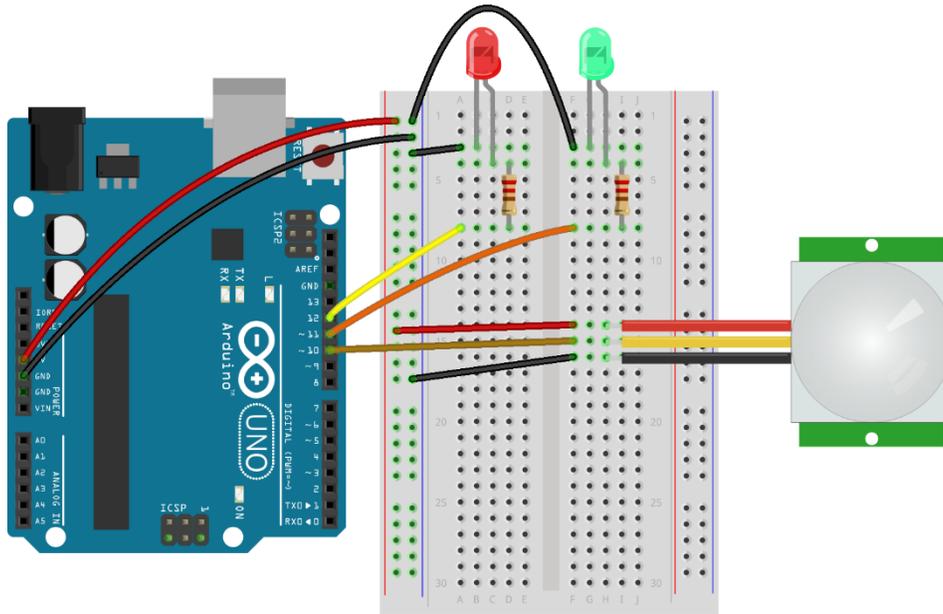
For this setup, we will need the following materials:

 <p>PIR Motion Sensor</p>	 <p>Arduino Uno R3</p>	 <p>USB A to USB B Cable</p>	 <p>Breadboard (Full-Size)</p>
 <p>Resistor – 330 Ω (1 pc)</p>	 <p>Connecting Wires – Male-Male 40 pcs</p>	 <p>Light Emitting Diode – Red (1 pc) Green (1 pc)</p>	

HARDWARE CONFIGURATION

Connect the line sensor to the Arduino Board according to the table below:

<u>Line/Obstacle Sensor</u>	--->	<u>Arduino</u>
VCC	--->	5V
GND	--->	GND
OUT	--->	DIGITAL PIN #8



fritzing

ARDUINO CODE

On a blank sketch in the Arduino IDE program, code the code below:

```

/*program for Motion Activated System*/

int PIRpin = 10; // input pin for arduino to read motion sensor logic high data
int greenLED = 11;
int redLED = 12;
int val;

void setup()
{
  Serial.begin(9600);
  pinMode(greenLED, OUTPUT);
  pinMode(redLED, OUTPUT);
  pinMode(PIRpin, INPUT);
}

void loop()
{
  val = digitalRead(PIRpin);
  if (val == LOW)
  {
    Serial.println("No Motion");
    digitalWrite(redLED, LOW);
  }
}

```

```
    digitalWrite(greenLED, HIGH);  
  }  
  else  
  {  
    Serial.println("Motion Detected!");  
    digitalWrite(redLED, HIGH);  
    digitalWrite(greenLED, LOW);  
  }  
  delay(1000);  
}
```

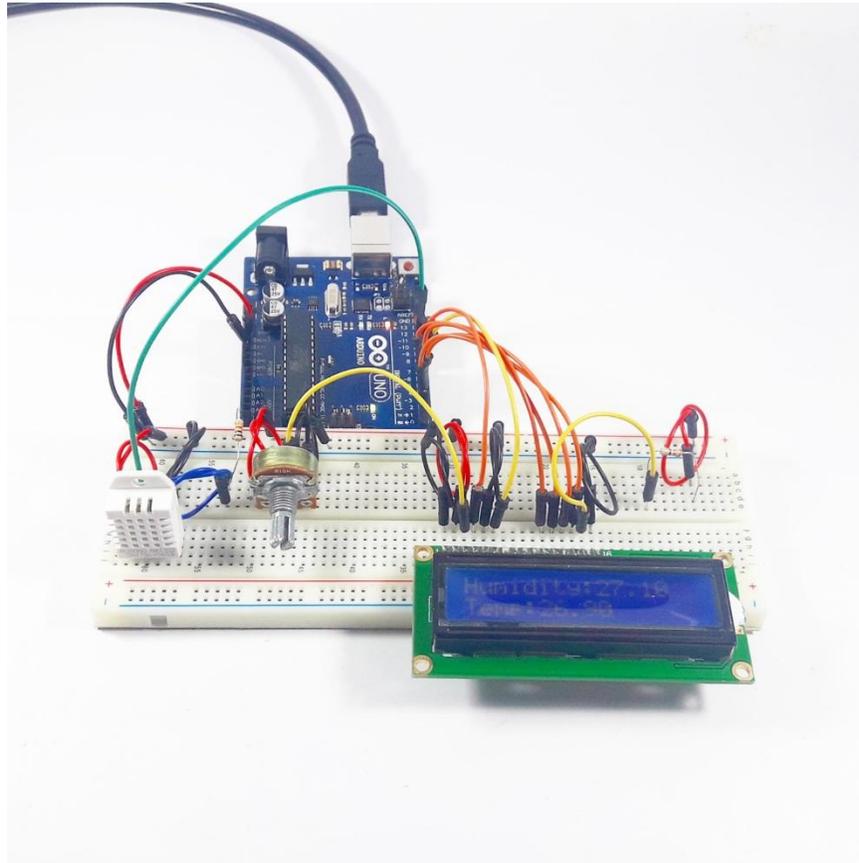
Compile the code, and upload it into the Arduino Board.

OUTPUT

After uploading the code open your Arduino Serial Terminal. It will display if motion is detected or no motion detected.

By placing and waving your hand or “body” in front of the motion sensor (with around 5 meter distance range in front of the sensor) the LED on the Arduino board will light up and in the Arduino Serial Terminal a “Motion Detected!” message will show. If no hand or body figure is moving in front of the sensor the LED on the Arduino will not light up and the Serial Terminal will display a “No Motion” message.

PROJECT 2: Digital Distance Measuring Device



The second project is an integration of two modules: the ultrasonic sensor and the LCD module. The ultrasonic sensor measures the distance from the obstruction while the LCD module displays the value of the distance measured.

PARTS LIST

For this setup, we will need the following materials:

			
LCD 2x16 Module	Arduino Uno R3	USB A to USB B Cable	Breadboard (Full-Size)

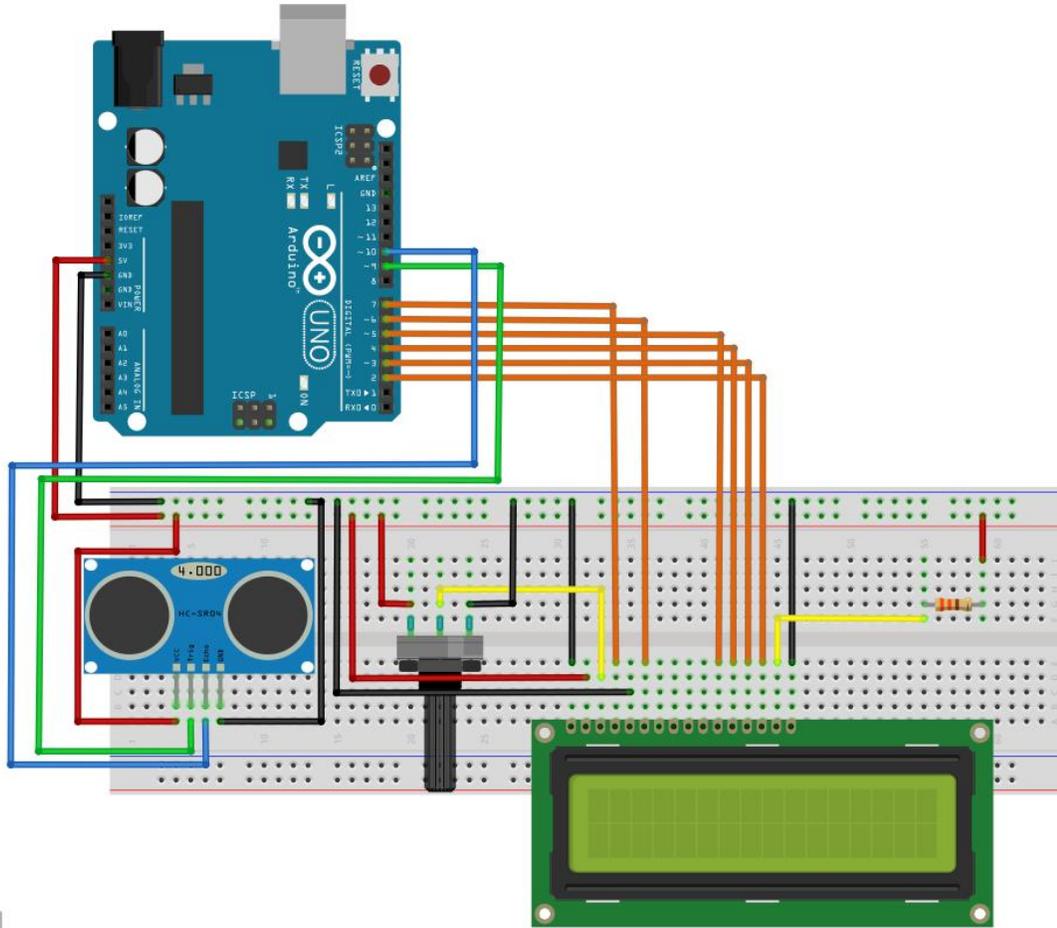
 <p>Resistor – 330 Ω (1 pc)</p>	 <p>Connecting Wires – Male-Male 40 pcs</p>	 <p>Potentiometer – 10 kΩ (1 pc)</p>	 <p>Ultrasonic Sensor HC-SR04</p>
---	--	--	--

WIRING CONNECTION

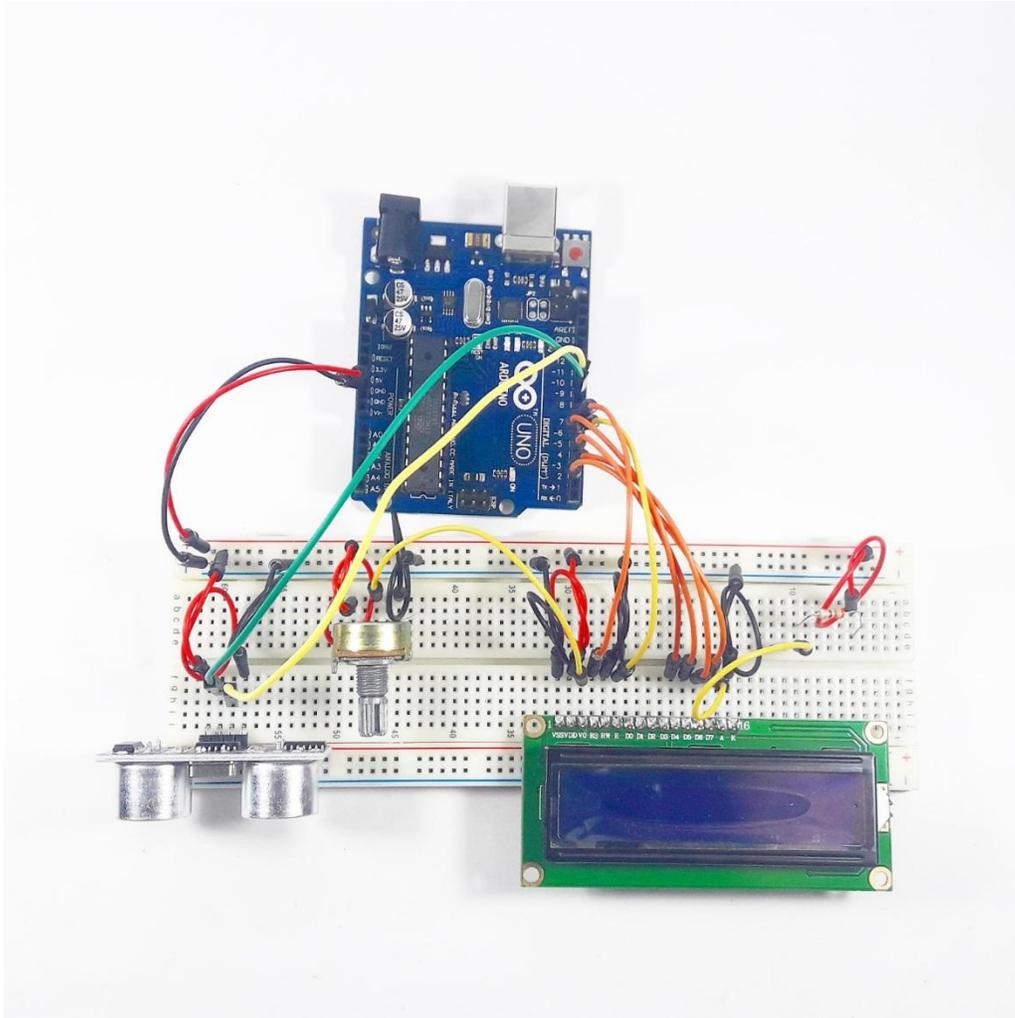
Setup the circuit as shown below:

LCD Module Connection	Ultrasonic Module Connection
LCD pin 1 (VSS) to GND	VCC to +5V
LCD pin 2 (VDD) to 5V+	Trig to Arduino pin 9
LCD pin 3 (V0) to Center of potentiometer	Echo to Arduino pin 10
LCD pin 4 (RS) to Arduino pin 7	GND to GND
LCD pin 5 (RW) to GND	
LCD pin 6 (E/Enable) to Arduino pin 6	
LCD pin 7-10 (D0-D3) no connection	
LCD pin 11 (D4) to Arduino pin 5	
LCD pin 12 (D5) to Arduino pin 4	
LCD pin 13 (D6) to Arduino pin 3	
LCD pin 14 (D7) to Arduino pin 2	
LCD pin 15 (A,Anode) to 300 ohm resistor to 5V+	
LCD pin 16 (K,Cathode) to GND	

*one side of potentiometer to 5V+ and one side to GND



fritzing



ARDUINO CODE

Open Arduino IDE. Copy the code below.

```

/*
 * Ultrasonic Sensor HC-SR04 interfacing with Arduino.
 */
#include <LiquidCrystal.h>

// defining the pins
const int trigPin = 9;
const int echoPin = 10;
// defining variables
long duration;
int distance;

LiquidCrystal lcd(7,6,5,4,3,2);
void setup() {
  pinMode(trigPin, OUTPUT); // Sets the trigPin as an Output
  pinMode(echoPin, INPUT); // Sets the echoPin as an Input
  lcd.begin(16,2);

```

```

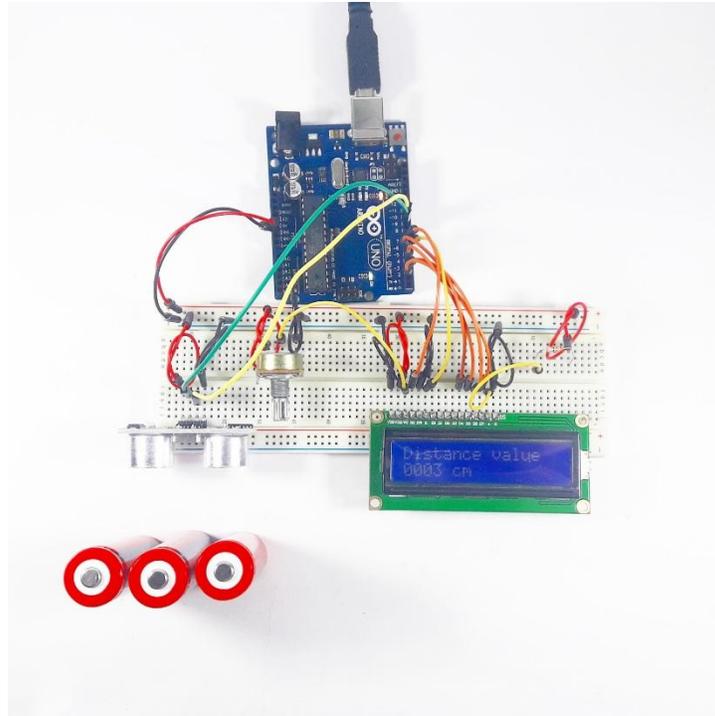
Serial.begin(9600);
lcd.print("Distance value");
}
void loop() {
// Clears the trigPin
//lcd.clear();
digitalWrite(trigPin, LOW);
delayMicroseconds(2);
// Sets the trigPin on HIGH state for 10 micro seconds
digitalWrite(trigPin, HIGH);
delayMicroseconds(10);
digitalWrite(trigPin, LOW);
// Reads the echoPin, returns the sound wave travel time in microseconds
//duration = pulseIn(echoPin, HIGH, 20000);
duration = pulseIn(echoPin, HIGH);
// Calculating the distance
distance= duration*0.034/2; //speed travels at 340 m/s or 0.034 cm/us, then sound
bounces back so distance should be divided into two.
// Prints the distance on the Serial Monitor
Serial.println(distance);
lcd.setCursor(0,1);
lcd.print("0000");
if (distance < 10)
{
  lcd.setCursor(3,1);
}
else if (distance >= 10 && distance < 100)
{
  lcd.setCursor(2,1);
}
else if (distance >= 100 && distance < 1000)
{
  lcd.setCursor(1,1);
}
else if (distance >= 1000)
{
  lcd.setCursor(0,1);
}
lcd.print(distance);
lcd.print(" cm");
delay(1000);
}

```

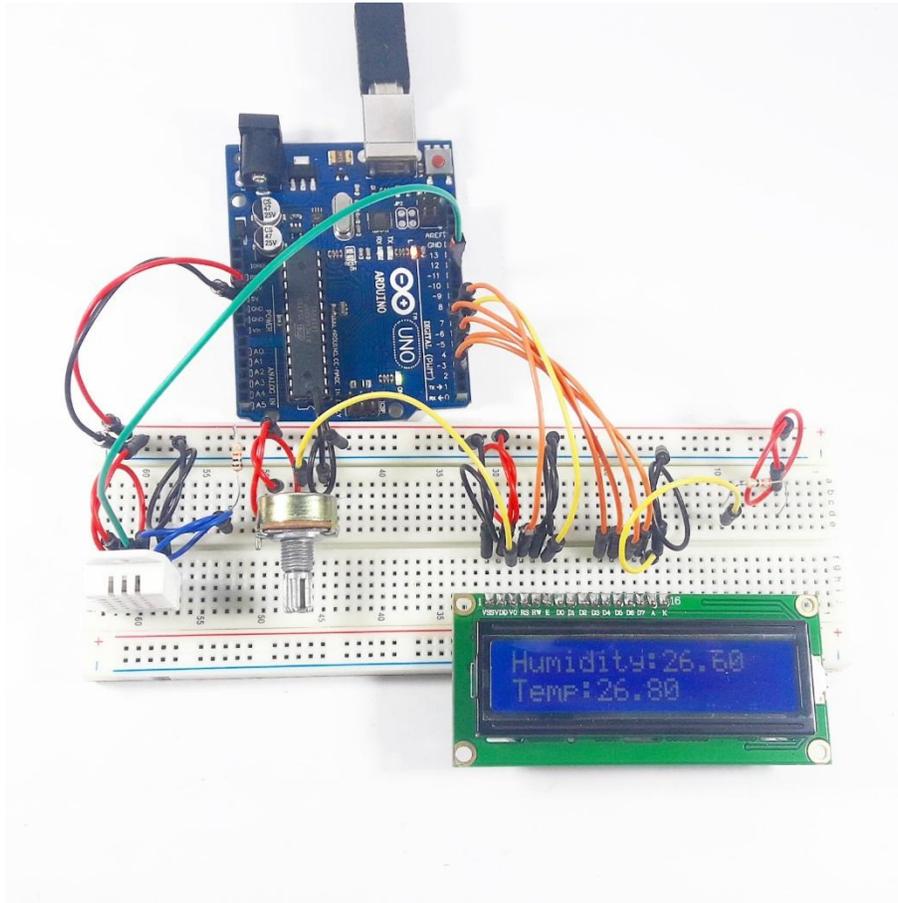
Check for errors in your code. Once done, compile and upload the code into your connected Arduino Uno board.

OUTPUT

After uploading, the LCD displays the distance of the object from the ultrasonic sensor in centimeters. You can change the contrast of the text by turning the knob of the potentiometer.



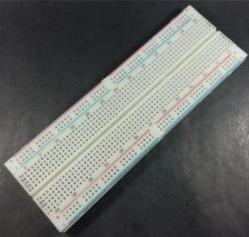
PROJECT 3: Temperature and Humidity Monitoring System

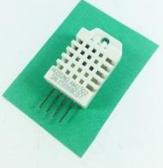


The last sample project is an integration of the DHT-22 temperature sensor, relay module (2 channel) and the LCD display module (2x16 characters). This project monitors the environment temperature, and displays the status on a display module. Additionally, when the measured temperature is above a certain threshold level, the relay switches on to run an external fan or connected device.

PARTS LIST

For this setup, we will need the following materials:

 <p>LCD 2x16 Module</p>	 <p>Arduino Uno R3</p>	 <p>USB A to USB B Cable</p>	 <p>Breadboard (Full-Size)</p>
--	---	--	---

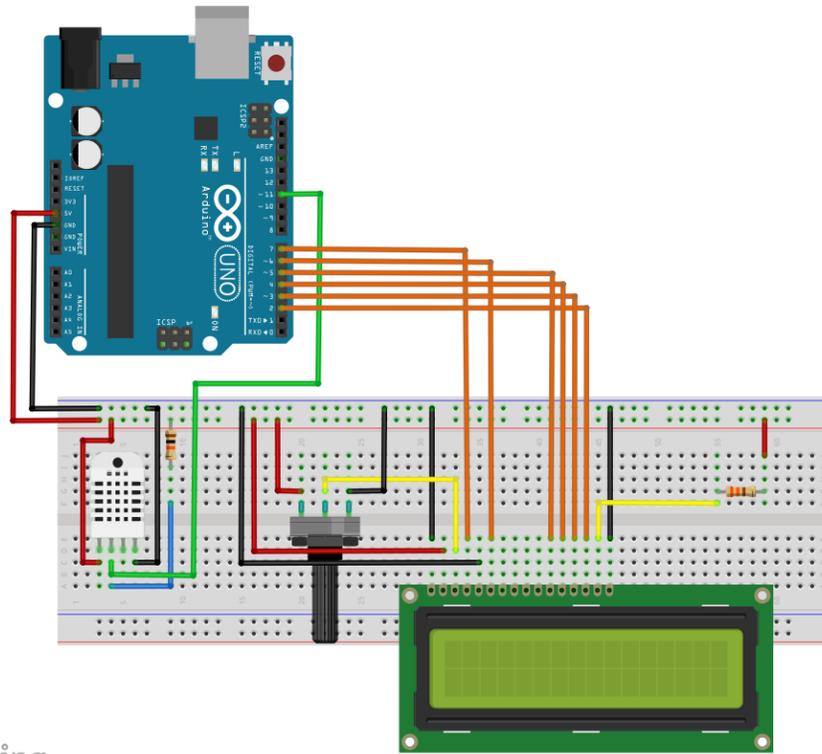
 <p>Resistor – 330 Ω (1 pc)</p>	 <p>Connecting Wires – Male-Male 40 pcs</p>	 <p>Potentiometer – 10 kΩ (1 pc)</p>	 <p>2 channel relay module (1 pc)</p>
 <p>DHT-22 Temperature and Humidity Sensor</p>	 <p>Connecting Wires – Male-Female 40 pcs</p>	 <p>Resistor – 10 kΩ (1 pc)</p>	

WIRING CONNECTION

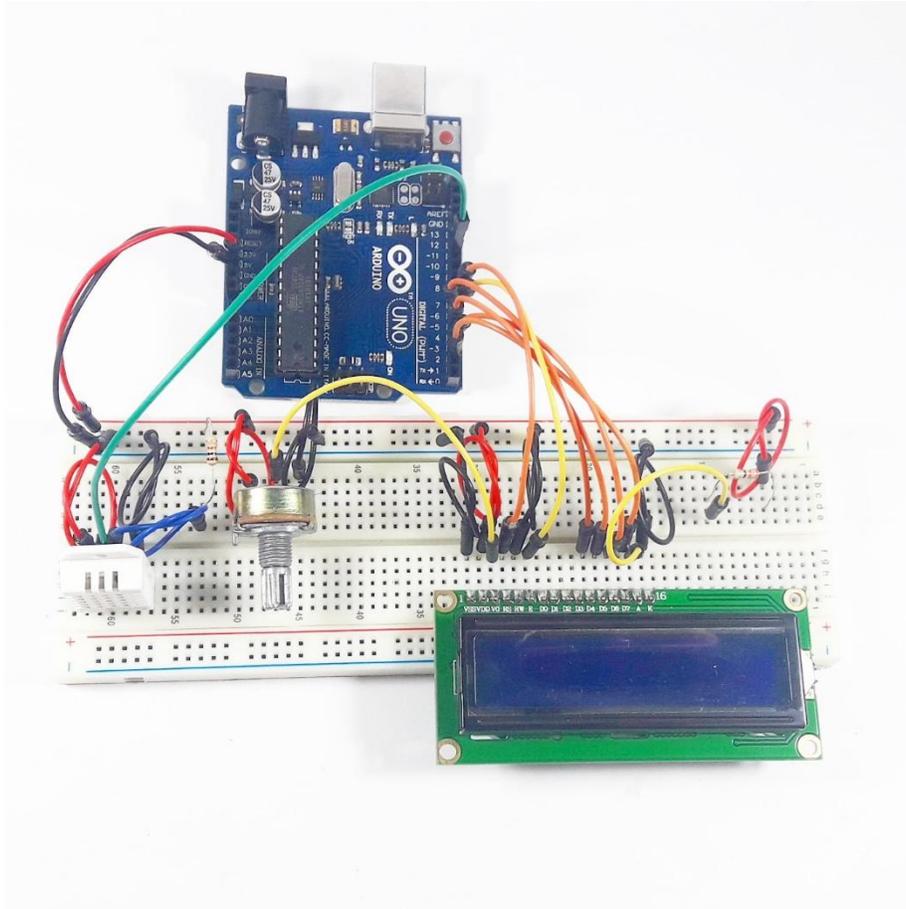
Follow the connection as shown in the image below.

LCD Module Connection	DHT-22 Module Connection
LCD pin 1 (VSS) to GND	VCC to +5V
LCD pin 2 (VDD) to 5V+	Data to Arduino pin 9 to 10k ohm resistor to 5V
LCD pin 3 (V0) to Center of potentiometer	GND to GND
LCD pin 4 (RS) to Arduino pin 7	
LCD pin 5 (RW) to GND	
LCD pin 6 (E/Enable) to Arduino pin 6	
LCD pin 7-10 (D0-D3) no connection	
LCD pin 11 (D4) to Arduino pin 5	
LCD pin 12 (D5) to Arduino pin 4	
LCD pin 13 (D6) to Arduino pin 3	
LCD pin 14 (D7) to Arduino pin 2	
LCD pin 15 (A,Anode) to 300 ohm resistor to 5V+	
LCD pin 16 (K,Cathode) to GND	

*one side of potentiometer to 5V+ and one side to GND



fritzing



ARDUINO CODE

Copy the code below into an empty Arduino IDE sketch.

```
#include "DHT.h"
#include <LiquidCrystal.h>

#define DHTPIN 11
#define DHTTYPE DHT22

LiquidCrystal lcd(7,6,5,4,3,2);
DHT dht(DHTPIN, DHTTYPE);

void setup() {
  // put your setup code here, to run once:
  dht.begin();
  lcd.begin(16,2);
}

void loop() {
  // put your main code here, to run repeatedly:
  float h = dht.readHumidity();
  float t = dht.readTemperature(); //in Celsius
```

```
lcd.setCursor(0,0);  
lcd.print("Humidity:");  
lcd.print(h);  
delay(10);  
lcd.setCursor(0,1);  
lcd.print("Temp:");  
lcd.print(t);  
delay(10);  
}
```

Check for errors in your code. Once done, compile and upload the code into your connected Arduino Uno board.

OUTPUT

After uploading, the LCD displays the temperature and humidity measured by the DHT-22 sensor. You can change the contrast of the text by turning the knob of the potentiometer.

