



## Keyes 24 in 1 Sensor Kit For Arduino



White to red Straw LED



Plug-in RGB Sensor



Hall Magnetic Sensor



Thermistors Sensor



Passive Buzzer module



Knock Sensor Module



Rotary encoder



18B20 Temperature Sensor



Piranha white to yellow LED



Photocell sensor



Adjustable potentiometer



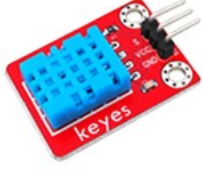
Infrared Obstacle Avoidance Sensor



Digital Push Button



Reed switch



DHT11 Temperature and Humidity Sensor



Laser Head Sensor



Line Tracking Sensor



3W LED Sensor



Microphone Sound Sensor



Active Buzzer Module



Digital Tilt Sensor



Crash Sensor



LM35 Temperature Sensor



5V Relay Module



## Catalog

1. Description.....	3
2. Kit List.....	4
4.Arduino IDE and Driver Installation.....	7
5. Projects.....	19
Project 1: LED Module.....	19
Project 2: Reed Module Module.....	22
Project 3: Active Buzzer Module.....	25
Project 4: Passive Buzzer Module.....	26
Project 5: Rotary Encoder Module.....	33
Project 6: Adjustable Potentiometer Module.....	37
Project 7: 5V 1-channel Relay Module.....	39
Project 8: Plug-in RGB Module.....	41
Project 9: Thermistors Sensor.....	44
Project 10: Button Sensor.....	46
Project 11: DHT11 Temperature and Humidity Sensor.....	49
Project 12: Photocell Sensor.....	52
Project 13: Tilt Sensor.....	55












Project 14: Microphone sound sensor.....	57
Project 15: Hall Magnetic Sensor.....	59
Project 16: Crash Sensor.....	62
Project 17: Knock Sensor.....	64
Project 18: Obstacle Avoidance Sensor.....	66
Project 19: LM35 Temperature Sensor.....	68
Project 20: Laser Head Sensor.....	71
Project 21: Line Tracking Sensor.....	73
Project 22: 18B20 Temperature Sensor.....	75
6. Resource.....	79

## **1. Description**











This kit contains 24 sensor modules, such as active buzzer module, 5V relay module, temperature and humidity sensor, etc. It is applicable for any kinds of microcontrollers and Raspberry Pi. Meanwhile, we have compiled projects for each sensor in the kit. Lets' get started!

## **2. Kit List**








Code	Name	Description	QTY	Picture
1	Keyes Module	keyes Straw White LED Module	1	
2	Keyes Module	keyes Reed Switch	1	
3	Keyes Module	keyes Piranha Yellow LED	1	
4	Keyes Module	keyes 3W LED Module	1	
5	Keyes Module	keyes Active Buzzer Module	1	
6	Keyes Module	keyes Passive Buzzer Module	1	
7	Keyes Module	keyes Rotary Encoder Module	1	
8	Keyes Module	keyes Adjustable Potentiometer Module	1	
9	Keyes Module	keyes 5V Relay Module	1	



10	Keys Module	keys Plug-in RGB Module	1	
11	Keys Sensor	keysThermistors Sensor	1	
12	Keys Sensor	keys Button Sensor	1	
13	Keys Sensor	keys DHT1 1Temperature and Humidity Sensor	1	
14	Keys Sensor	keys Photocell Sensor	1	
15	Keys Sensor	keys Tilt Sensor	1	
16	Keys Sensor	keys Microphone Sound Sensor	1	
17	Keys Sensor	keys Hall Magnetic Sensor	1	
18	Keys Sensor	keys Crash Sensor	1	
19	Keys	keys Knock	1	



	Sensor	Sensor		
20	Keyes Sensor	keyes Obstacle Avoidance Sensor	1	
21	Keyes Sensor	keyes LM35 Temperature Sensor	1	
22	Keyes Sensor	keyes Laser Head Sensor	1	
23	Keyes Sensor	keyes Line Tracking Sensor	1	
24	Keyes Sensor	keyes DS18B20 Temperature Sensor	1	

### 3. Arduino IDE and Driver Installation

When getting the development board, first of all you have to install the Arduino IDE and the driver, and all relevant files can be found on the official website. The following link includes various systems, various versions of the Arduino



IDE and drivers whatever you choose.

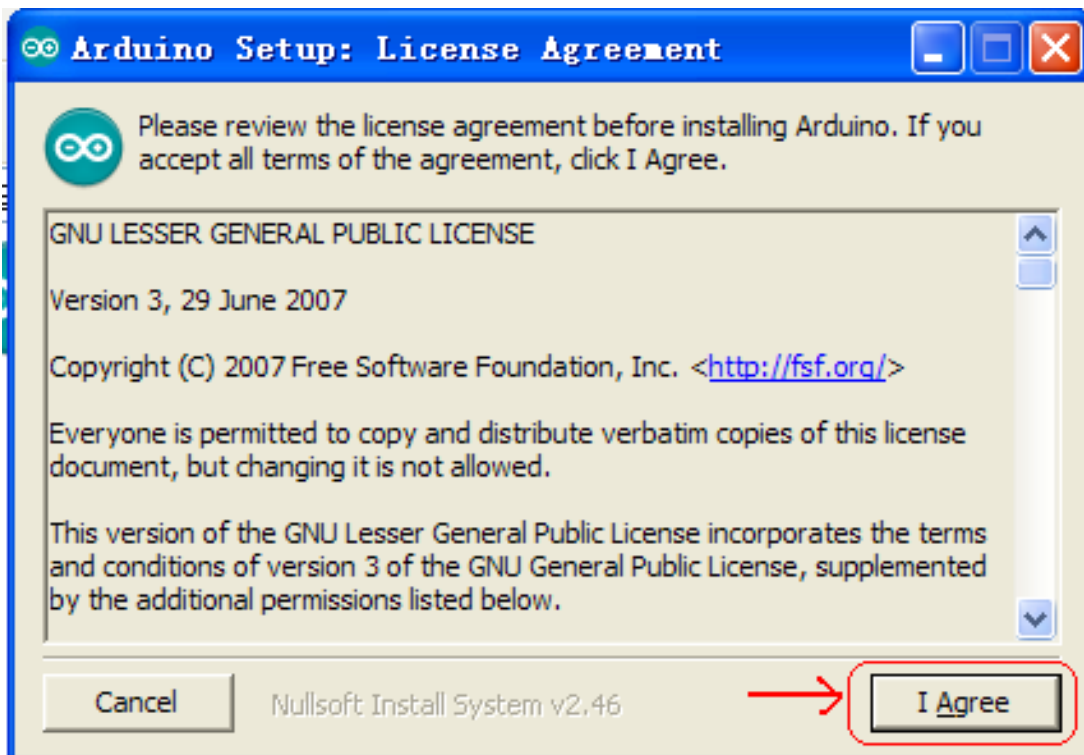
<https://www.arduino.cc/en/Main/OldSoftwareReleases#1.5.x>

Next, we first introduce the installation method of Arduino IDE-1.5.6 version in the Windows system.

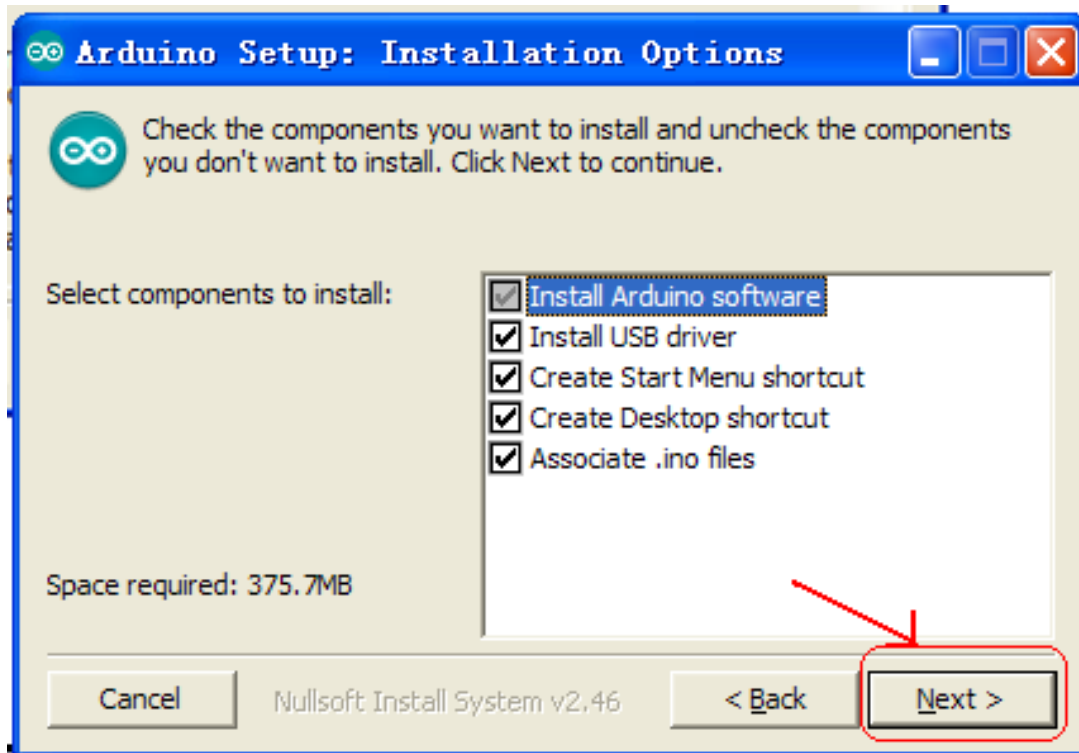
The file downloaded is an **arduino-1.5.6-r2-windows.zip** compression folder, please unzip it to the hard disk.

Double-click Arduino-1.5.6 .exe file.

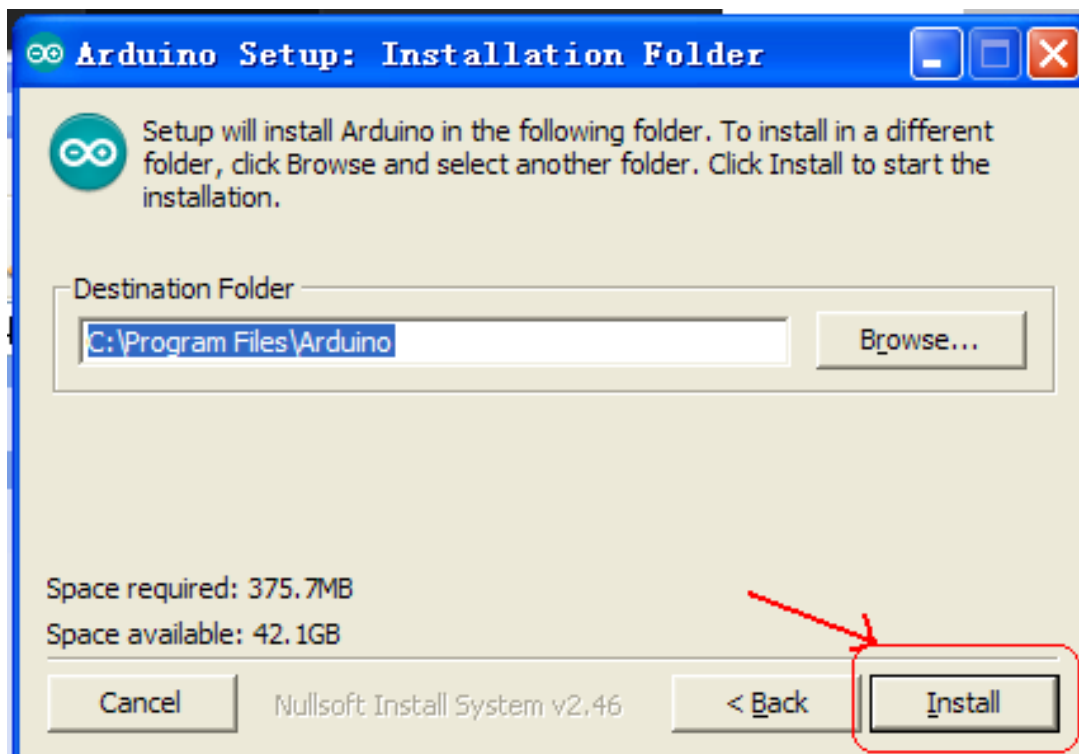
Please refer to the following setup figures:



Click "I Agree". Then, click "Next"

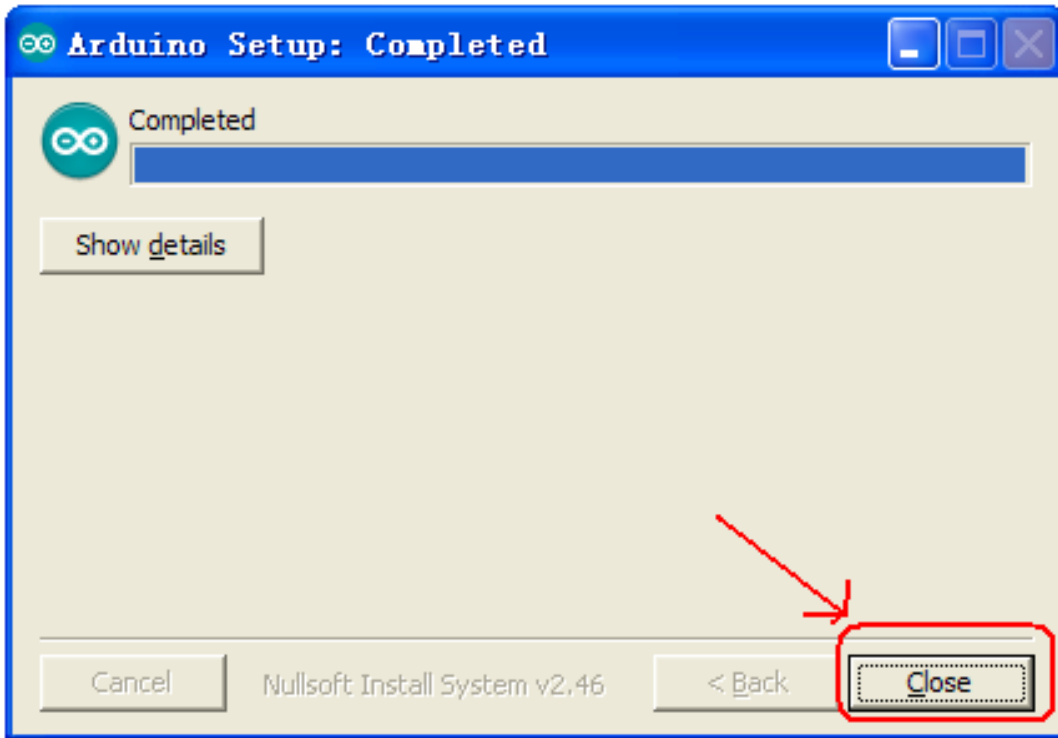


Next, click "Install".

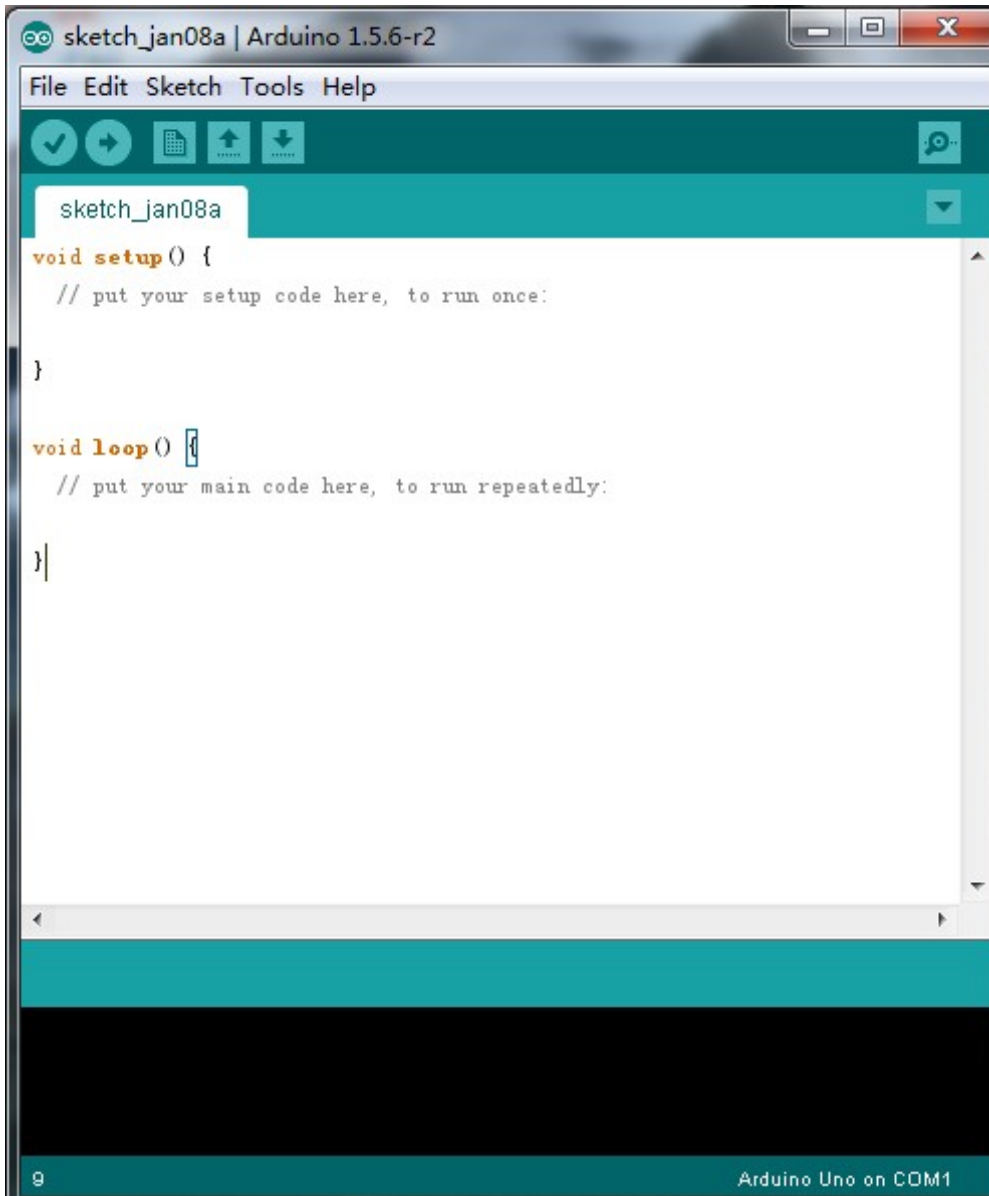


Finally, click "Close" after completing the installation.





The figure below shows the interface of Arduino1.5.6 version:



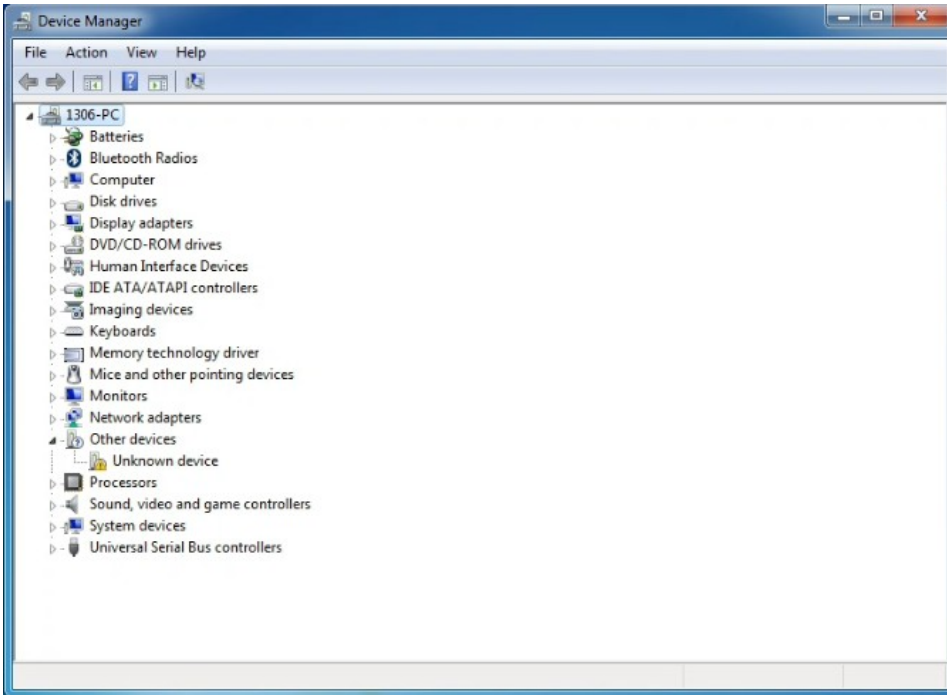
Next, we will introduce the driver installation of keyestudio UNO R3 development board.

Let's move on to the driver installation in the WIN 7 system.

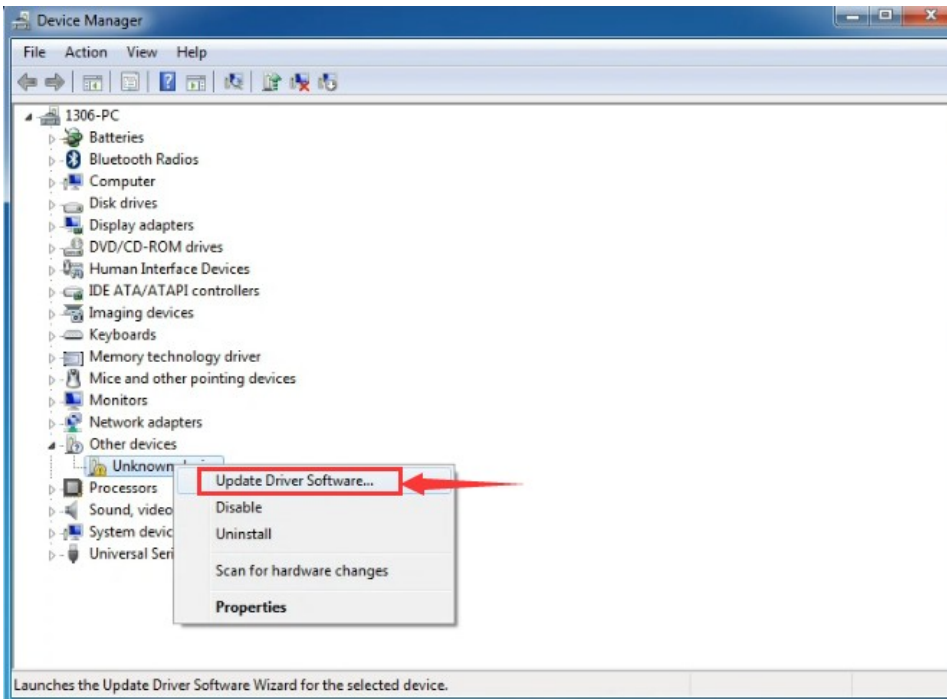
A. When you connect UNO board to your computer at the first time, right click "Computer" —>"Properties"—> "Device manager", you can see "Unknown



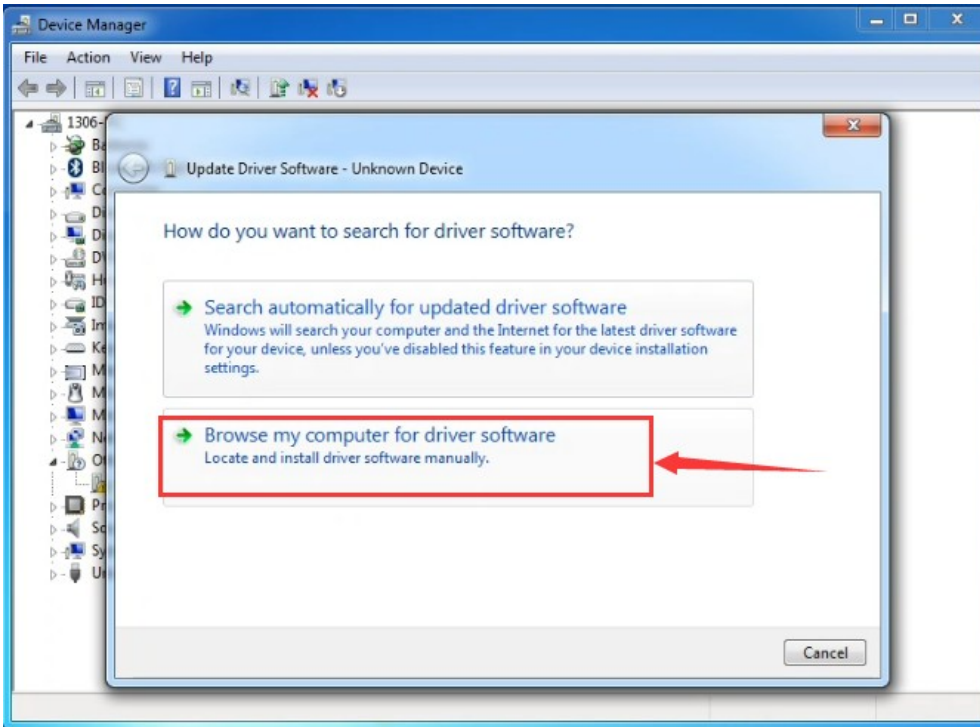
devices”.



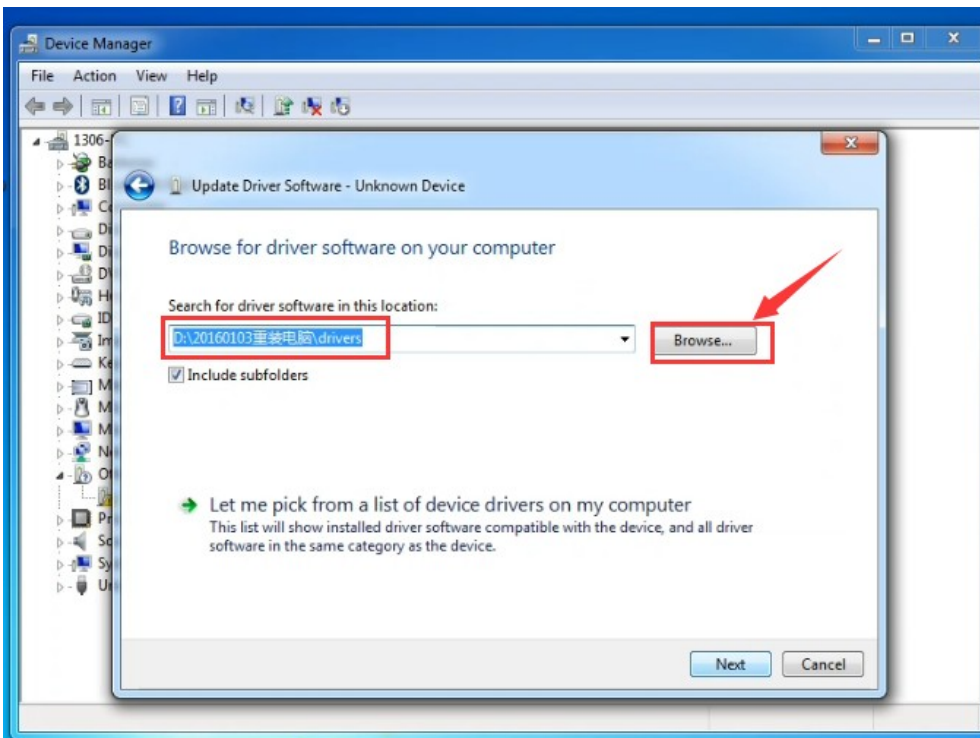
B. Click “Unknown devices”, select “Update Driver software”.



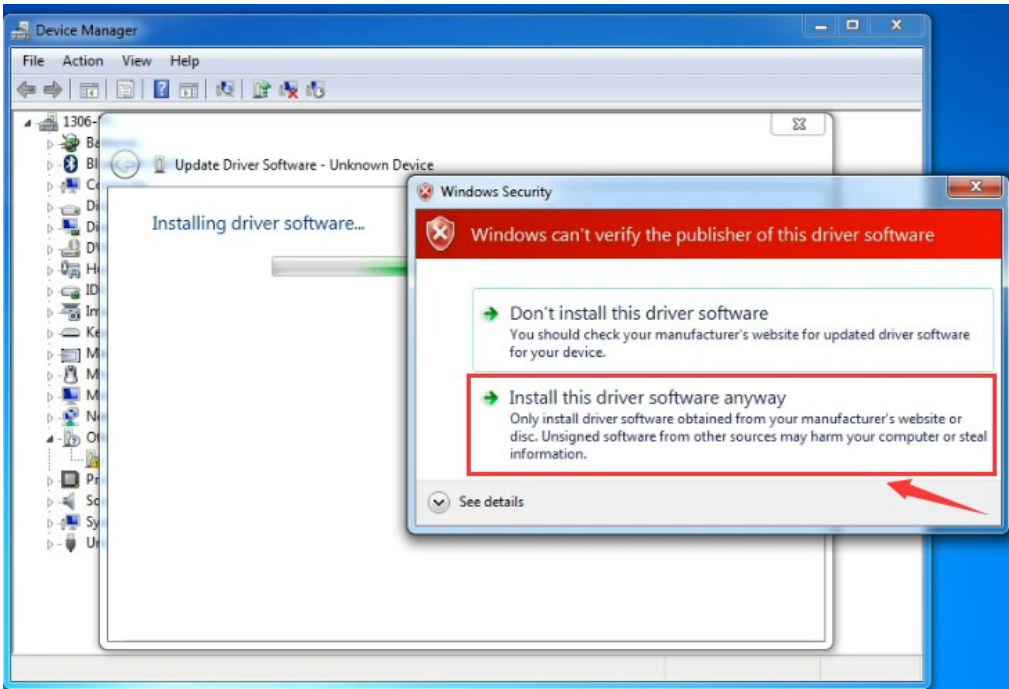
C. In this page, click “Browse my computer for driver software”.



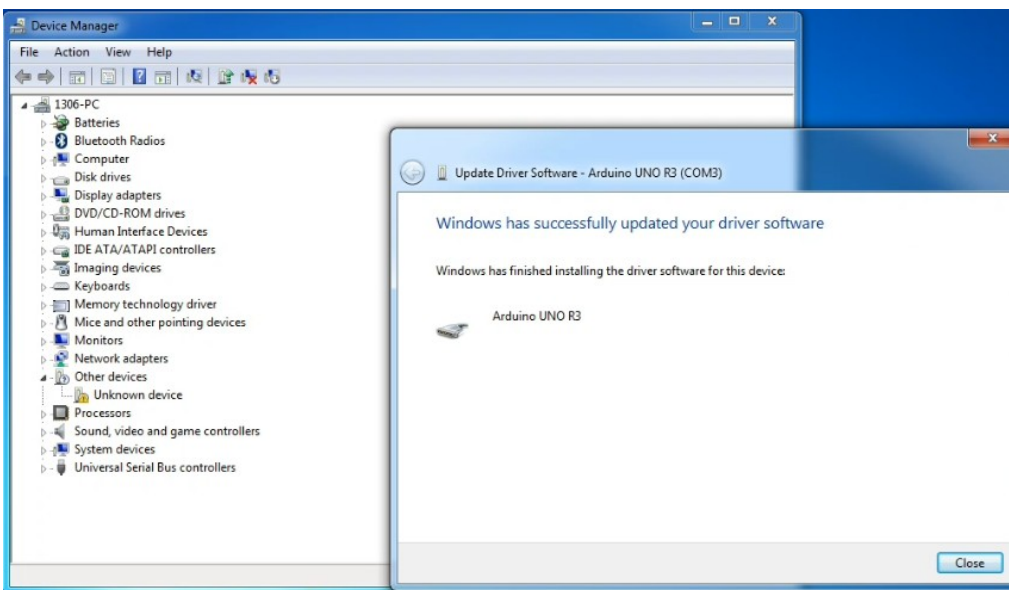
D. Find the "drivers" folder.



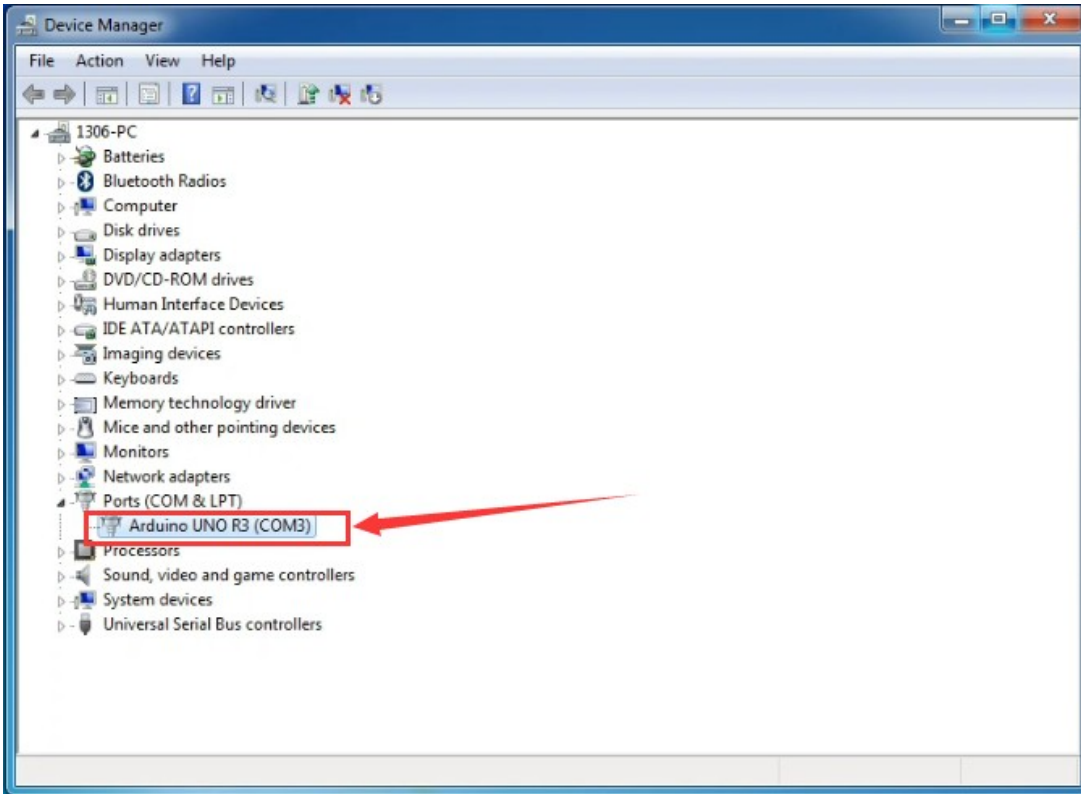
E. Click "Next"; select "Install this driver software anyway" to begin the installation.



F. Installation completed, click "Close".



G. After installation, go to see the "Device manager" again. right click "Computer" —> "Properties"—> "Device manager", you can see the device as below figure shown.



## 1. Using Method of Arduino IDE

When successfully installing the USB driver for UNO R3 development board, you can find the corresponding serial port in Windows Device Manager.

Next, we will show you the first program showing the "Hello World!" on the serial monitor.

### Sample Code as below:

```
////////////////////////////////////
```

```
int val;
```

```
int ledpin=13;
```

```
void setup()
```

```
{
```

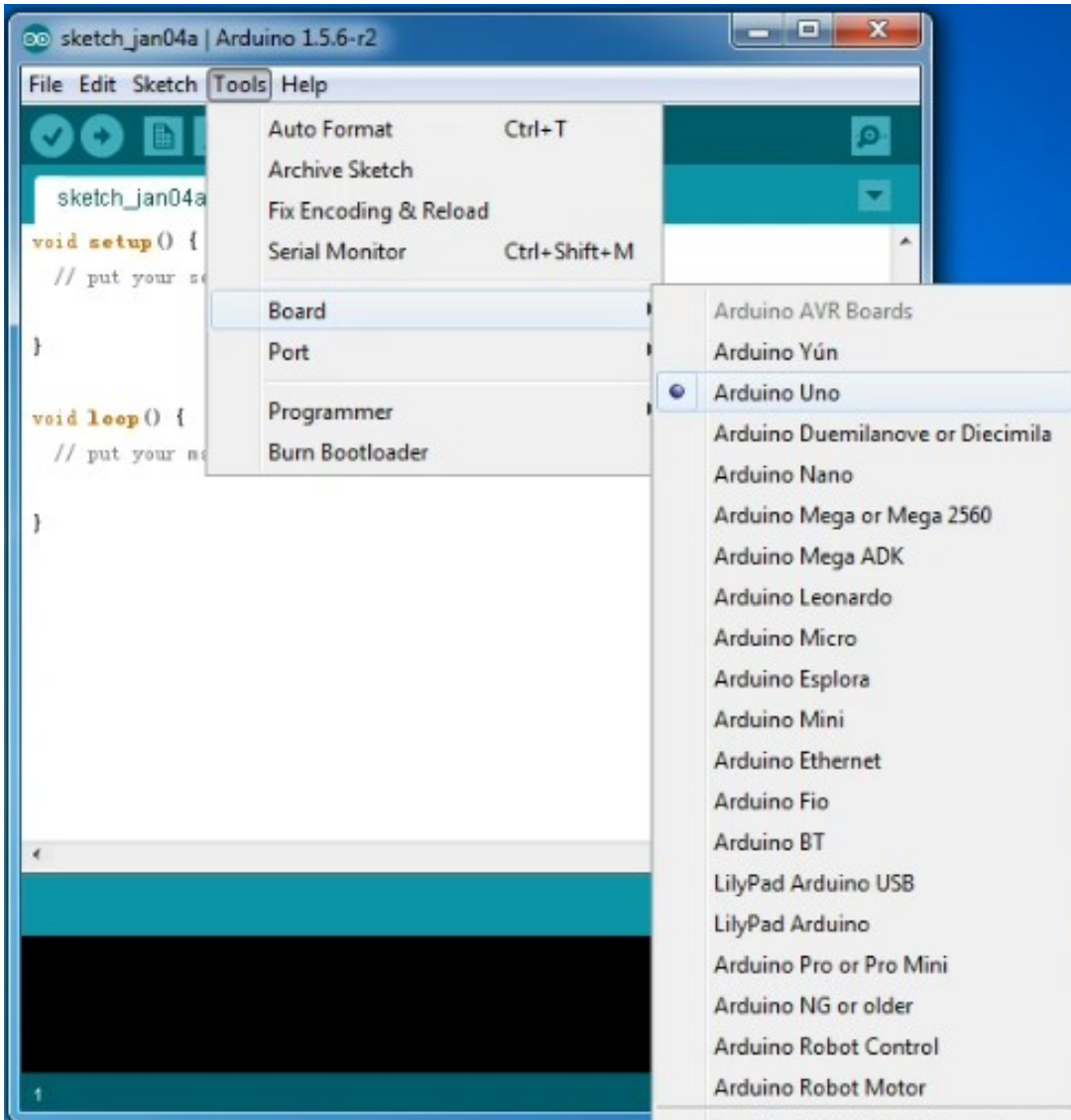


```
Serial.begin(9600);  
  
pinMode(ledpin,OUTPUT);  
  
}  
  
void loop()  
  
{  
  
val=Serial.read();  
  
if(val=='R')  
  
{  
  
digitalWrite(ledpin,HIGH);  
  
delay(500);  
  
digitalWrite(ledpin,LOW);  
  
delay(500);  
  
Serial.println("Hello World!");  
  
}  
  
}  
  
////////////////////////////////////
```

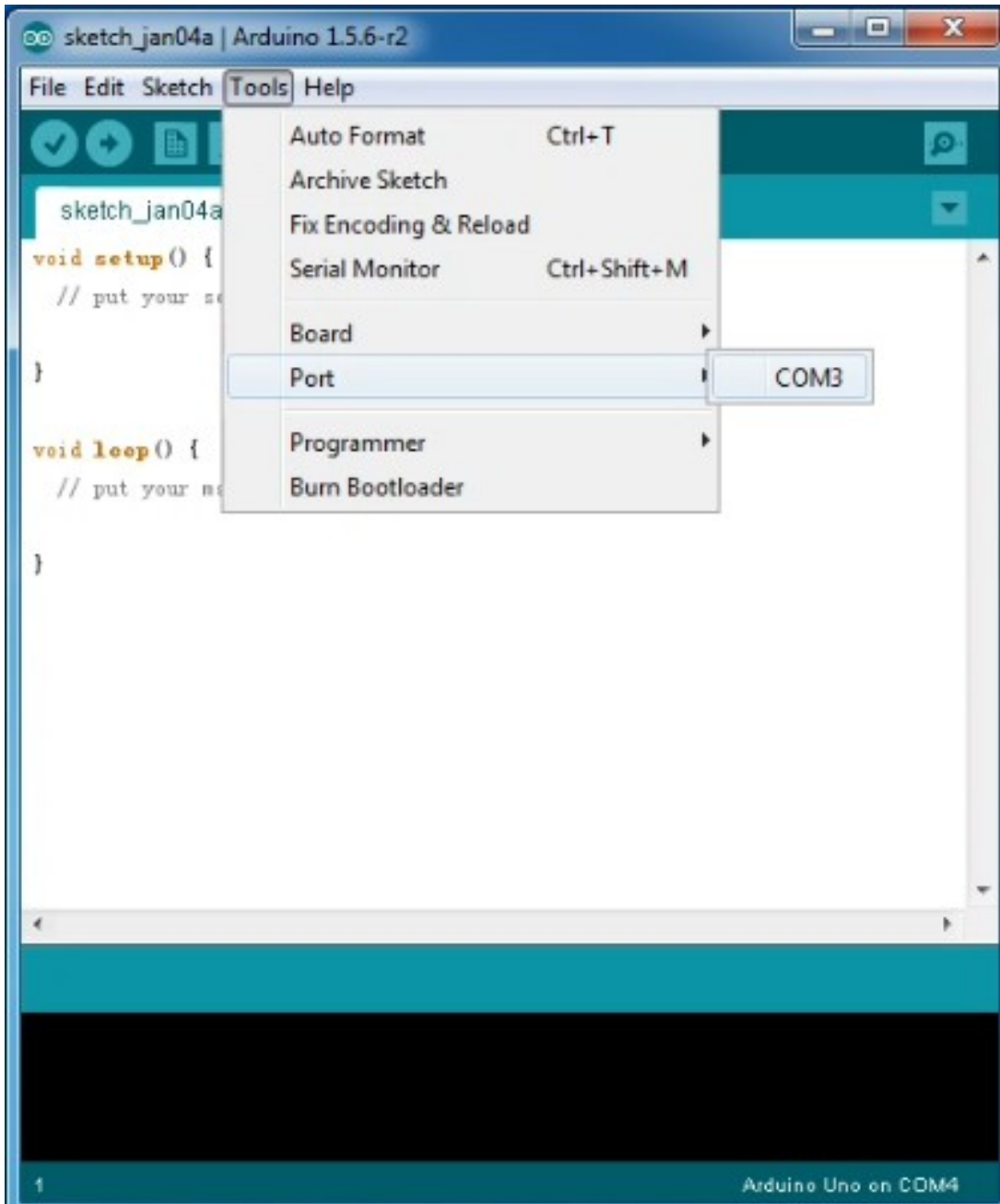
Open Arduino software, print the program to make UNO R3 development board display the character "Hello World! ". When the board receives the instruction, D13 indicator light on the board blinks and "Hello World! " is displayed on the monitor.



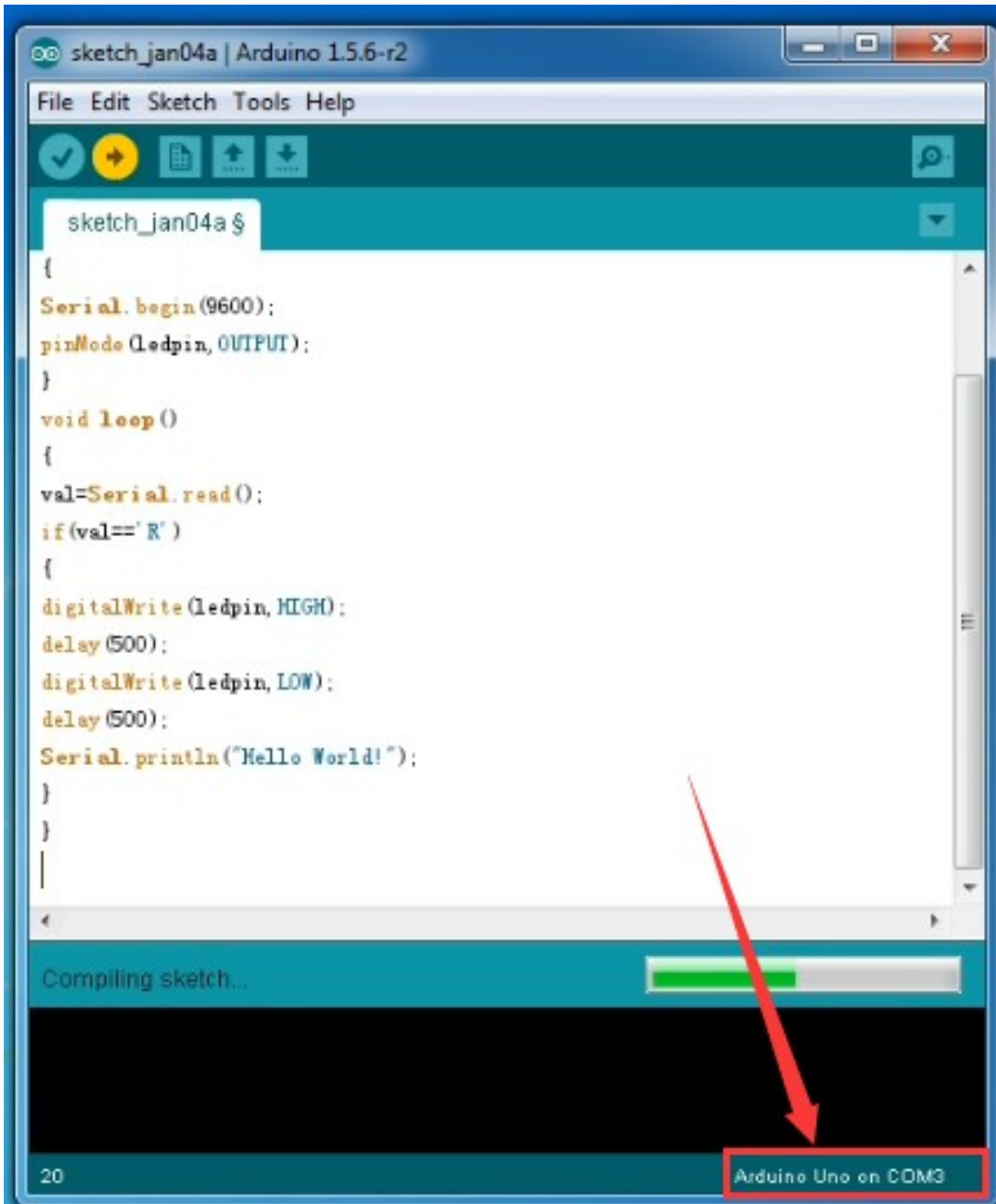
First set the Board and COM port, shown below.







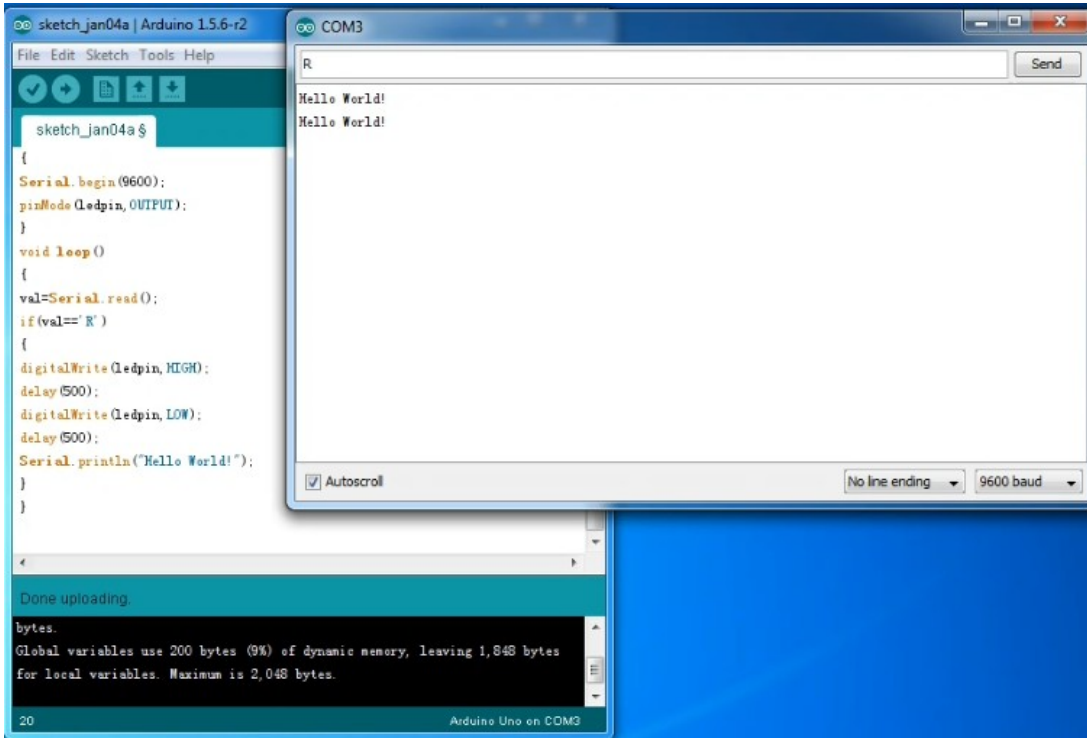


If setting well the board and port, you can see the display on the bottom right corner, which is the same as the Device Manager display.



Then, click the verify  to compile the sketch, if no mistake, click upload  to upload the program.

Done uploading, open the serial monitor, print an "R" and click "Send", you can see the D13 indicator on the UNO R3 development board blinks once, and the "Hello World!" is displayed on the serial monitor. Shown below.



Congrats. Your first programming is done well!

## 4. Projects

### Project 1: LED Module

#### Description

In this project, we mainly test LED module. Connect the signal end of LED module to digital 3(PWM port) of the development board. We make two experiments to test, one is to blink LED, another one is to control LED brightness via PWM port, make LED become brighter and darker to simulate the human breath.



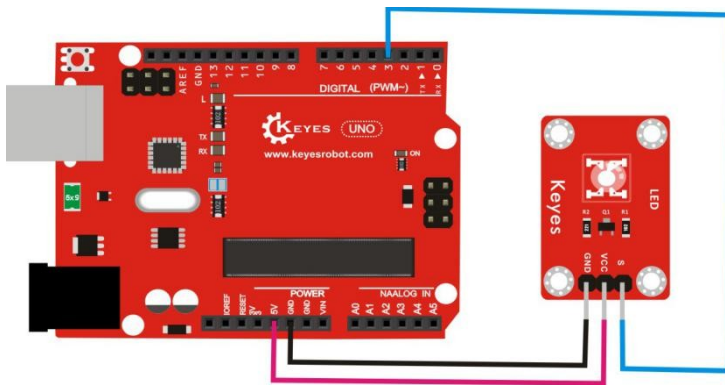
## Equipment

Development board \* 1

USB cable \* 1

LED module \* 1 (straw hat LED, Piranha LED module and 3W LED module)

DuPont Lines



### Test Code A:

```
int led = 3;           //Define digital port 3

void setup()
{
  pinMode(led, OUTPUT); //set LED to output
}

void loop()
{
```



```
digitalWrite(led, HIGH); //turn on led
delay(1000); //delay in 1000ms
digitalWrite(led, LOW); //turn off led
delay(1000); //delay in 1000ms
}
```

### **Code B:**

```
int ledPin = 3; // Define digital port 3
void setup()
{
pinMode(ledPin, OUTPUT); // set ledPin to output}
void loop()
{
for (int a=0; a<=255;a++) // LED is gradually brightening
{
analogWrite(ledPin,a); // turn on led, adjust the brightness, in the range 0-
255, led is brightest when at 255
delay(10); // delay in 0.01s
}
for (int a=255; a>=0;a--) // LED is gradually darkening
{
analogWrite(ledPin,a); // turn on led, adjust the brightness, in the range 0-
255, led is brightest when at 255
```



```
delay(10); // delay in 0.01s  
}  
delay(1000); // delay in 1s  
}
```

## **Test Result**

After upload code A and power on, we can see that the LED blinking with the interval of 1s. Upload code B and power up, we can see that the LED gradually brightens, then darkens, and alternately.

## **Project 2: Reed Module Module**

### **Description**

This module is mainly composed of a reed switch. After the module is connected to the power supply, the signal terminal outputs high, and the LED on the sensor gets dark. On the contrast, when the magnetic field is added at the signal end, low level will be outputted and LED will get brighter. In the project, we control the D13 of Arduino UNO board with sensor.



## Equipment

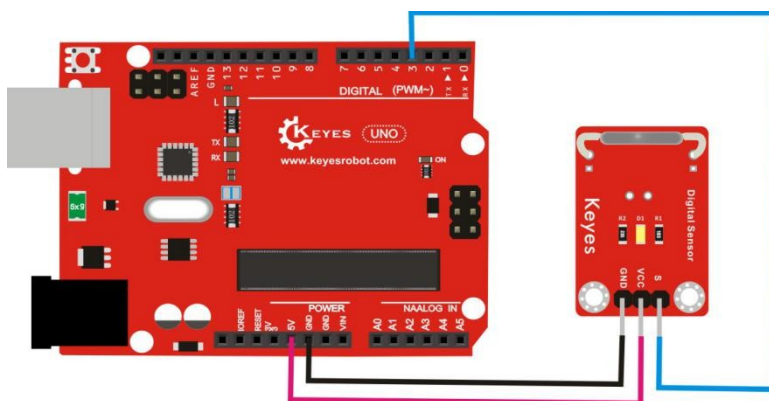
Development board \* 1

USB cable \* 1

Reed switch module \* 1

DuPont Lines

## Connection Diagram





## Test Code

```
int Led=13;//Define digital port 13

int buttonpin=3; //Define digital port 3

int val;//Define digital variables val

void setup()

{

pinMode(Led,OUTPUT);//set Led to output

pinMode(buttonpin,INPUT);//set buttonpin to input }

void loop()

{

val=digitalRead(buttonpin);// read the value of digital 3, assign the value

of val

if(val==LOW)//when val is HIGH level

{

digitalWrite(Led,HIGH); //LED lights up}

else

{

digitalWrite(Led,LOW); //LED is off

}

}
```





## **Test Result**

Wire according to the above figure and upload the code. After powering on, the D13 indicator on the Arduino UNO board and the D1 indicator go off. When a magnet is near the module, the D13 and D1 light on.

## **Project 3: Active Buzzer Module**

### **Description**

It is mainly made up of an active buzzer, this sensor is an integrated electronic sounder by a DC power supply.

After powering on, the signal end is inputted high level, then the buzzer sounds. In the project, we make active buzzer turn on and off.

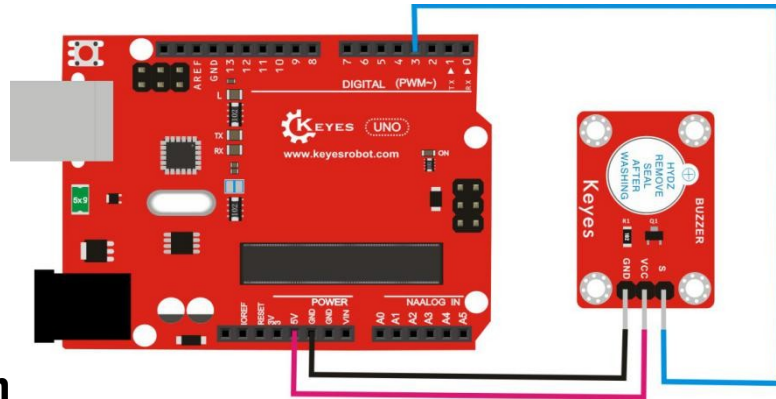
### **Equipment**

Development board \* 1

USB cable \* 1

Active buzzer module \* 1

DuPont Lines



## Connection Diagram

## Test Code

```
int buzzPin = 3; //Define digital port 3

void setup()
{
  pinMode(buzzPin, OUTPUT); //set buzzPin to output}
void loop()
{
  digitalWrite(buzzPin, HIGH); //active buzzer sounds
  delay(2000); //delay in 2s
  digitalWrite(buzzPin, LOW); //active buzzer turns off
  delay(2000); //delay in 2s
}
```

## Test Result

The active buzzer can make a sound with a high-level buzzer. Upload the code, wire and power on, the active buzzer will sound for 2s, mute for 2s and alternately.



## **Project 4: Passive Buzzer Module**

### **Description**

Buzzer is classified as active buzzer and passive buzzer, whereas the passive buzzer must be driven by square waves due to no oscillation source inside.

In the experiment, we connect signal end of passive buzzer to the digital 3 of development board. The passive buzzer is driven by square waves produced from digital 3. There are two experiments made to test, one is to alternately output the square waves with two kind of frequency at the digital 3, the passive buzzer will sound; another is to generalize the buzzer to play "Ode to Joy" with all kinds of frequency at digital 3 port and beats set.

### **Equipment**

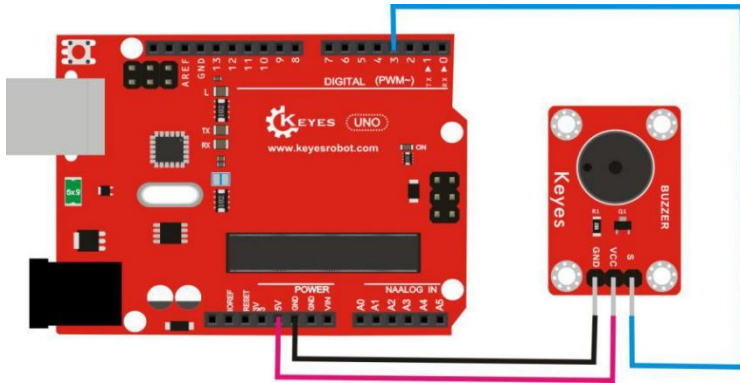
Development board \* 1

USB cable \* 1

Passive buzzer module \* 1

DuPont Lines

### **Connection Diagram**



### Test Code A:

```
int buzzer=3;      //Define digital port 3

void setup()
{
  pinMode(buzzer,OUTPUT);//SET buzzer to output}

void loop()
{
  unsigned char i,j;//Define variables i, j
  while(1)
  {
    for(i=0;i<80;i++)// output the sound with frequency
    {
      digitalWrite(buzzer,HIGH);
      delay(1);//delay in 1ms
      digitalWrite(buzzer,LOW);
      delay(1);//delay in 1ms
    }
  }
}
```



```
for(i=0;i<100;i++)// output the sound with another frequency
{
digitalWrite(buzzer,HIGH);
delay(2);//delay in 2ms
digitalWrite(buzzer,LOW);
delay(2);//delay in 2ms
}
}
}
```

**Code B:**

```
#define D0 -1
#define D1 262
#define D2 293
#define D3 329
#define D4 349
#define D5 392
#define D6 440
#define D7 494
#define M1 523
#define M2 586
#define M3 658
```



```
#define M4 697
#define M5 783
#define M6 879
#define M7 987
#define H1 1045
#define H2 1171
#define H3 1316
#define H4 1393
#define H5 1563
#define H6 1755
#define H7 1971
//List all frequency of D
#define WHOLE 1
#define HALF 0.5
#define QUARTER 0.25
#define EIGHTH 0.25
#define SIXTEENTH 0.625
//list all beats
int tune[]= //list all frequency according to the notation
{
    M3,M3,M4,M5,
    M5,M4,M3,M2,
```



M1,M1,M2,M3,

M3,M2,M2,

M3,M3,M4,M5,

M5,M4,M3,M2,

M1,M1,M2,M3,

M2,M1,M1,

M2,M2,M3,M1,

M2,M3,M4,M3,M1,

M2,M3,M4,M3,M2,

M1,M2,D5,D0,

M3,M3,M4,M5,

M5,M4,M3,M4,M2,

M1,M1,M2,M3,

M2,M1,M1

};

float durt[]= //List beats based on notation

{

1,1,1,1,

1,1,1,1,

1,1,1,1,

1+0.5,0.5,1+1,

1,1,1,1,



```
1,1,1,1,  
1,1,1,1,  
1+0.5,0.5,1+1,  
1,1,1,1,  
1,0.5,0.5,1,1,  
1,0.5,0.5,1,1,  
1,1,1,1,  
1,1,1,1,  
1,1,1,0.5,0.5,  
1,1,1,1,  
1+0.5,0.5,1+1,  
};  
int length;  
int tonepin=3; // Interface 3 is required  
void setup()  
{  
    pinMode(tonepin,OUTPUT);  
    length=sizeof(tune)/sizeof(tune[0]); //calculate the length  
}  
void loop()  
{  
    for(int x=0;x<length;x++)
```





```
{  
    tone(tonopin,tune[x]);  
        delay(500*durt[x]); //adjust delay according to beats , number 500  
can be changed freely, 500 is more applicable  
    noTone(tonopin);  
}  
delay(2000);  
}
```

## Test Result

After upload code A, the passive buzzer will emit two different sounds, and alternately. After upload code B and power on, the passive buzzer will play the song of "Ode to Joy".

## Project 5: Rotary Encoder Module

### Description

This module is mainly composed of a rotary encoder.

It can count the number of output pulses by rotating clockwise and



counterclockwise. This rotation count is unlimited, reset to the initial state, that is, counting from 0. In the project, we use a rotary encoder module to control the on and off of two straw LEDs.

## Equipment

Development board \* 1

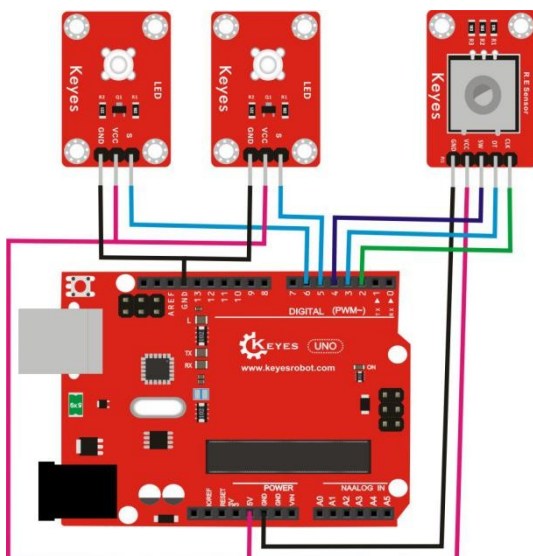
USB cable \* 1

Rotary encoder module \* 1

LED module \* 2

DuPont Lines

## Connection Diagram



## Test Code

```
const int interruptA = 0; //interrupt0 is at digital 2
```



```
const int interruptB = 1;//interrupt1 is at digital 3

int CLK = 2; // Define digital port 2

int DAT = 3; // Define digital port 3

int BUTTON = 4; // Define digital port 4

int LED1 = 5; // Define digital port 5

int LED2 = 6; // Define digital port 6

int COUNT = 0;//set digital variable COUNT to 0

void setup()

{

attachInterrupt(interruptA, RoteStateChanged, FALLING);

//When high level inverts into low level at digital 2, the trigger stops

pinMode(CLK, INPUT);//set CLK to input

digitalWrite(2, HIGH); // set digital 2 to HIGH level

pinMode(DAT, INPUT); //sets DAT to input

digitalWrite(3, HIGH); //set digital 3to HIGH level

pinMode(BUTTON, INPUT); //set BUTTON input

digitalWrite(4, HIGH); //set digital 4 to HIGH level

pinMode(LED1, OUTPUT);//set LED1 to output

pinMode(LED2, OUTPUT);//set LED1 to output

Serial.begin(9600); //set to baud rate

}

void loop()
```



```
{
if (digitalRead(BUTTON)==LOW)//when low level is at digital 4
{
COUNT = 0; //set COUNT to 0
Serial.println("STOP COUNT = 0");//the "shown" corresponds to
contentdigitalWrite(LED1, LOW);//LED1 gets dark
digitalWrite(LED2, LOW);//LED2 gets dark
delay (2000);//delay in 2S
}
Serial.println(COUNT);//display COUNT data
}
void RoteStateChanged() //when high level changes into low level at digital
2
{
if (digitalRead(DAT)==HIGH) // when high level is at digital 3
{
COUNT++;//digital variable COUNT plus 1
digitalWrite(LED1, HIGH);//LED1 lights up
digitalWrite(LED2, LOW);//LED2 gets dark
delay(200);//delay in 0.2S
}
}
else
```



```
{  
COUNT--;//digital variables COUNT deducts 1  
digitalWrite(LED2, HIGH);//LED2 lights on  
digitalWrite(LED1, LOW);//LED1 gets dark  
delay(200);//delau in 0.2S  
}  
}
```

### **Test Result**

Upload the code, wire and power on, we can control the two straw LEDs on and off by rotating the rotary encoder.

## **Project 6: Adjustable Potentiometer Module**

### **Description**

This module mainly consists of adjustable potentiometer. After power on, the analog input value can be adjusted by rotating the potentiometer on the module. In the project, connect the S end of module to the A0 port of development board, then the analog value is shown on the serial monitor.



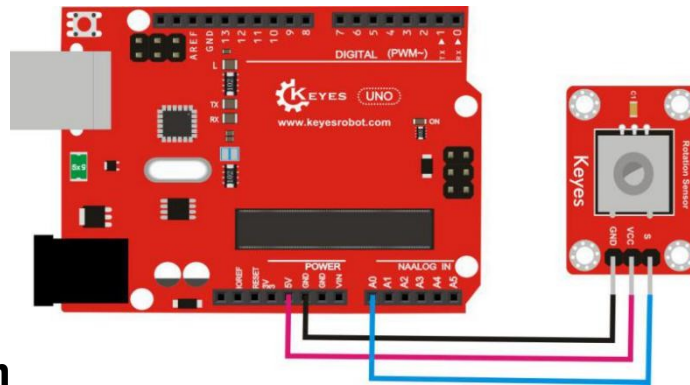
## Equipment

Development board \* 1

USB cable \* 1

Adjustable potentiometer module \* 1

DuPont Lines



## Connection Diagram

**Test Code**  
`int sensorPin = A0 ; //Define analog port A0`

`int value = 0; //set value to 0`

`void setup()`

`{`

`Serial.begin(9600); //Set the baud rate`

`}`

`void loop()`

`{`

`value = analogRead(sensorPin); //set value to reading the value of A0`

`Serial.println(value, DEC); //display value, and wrap word`

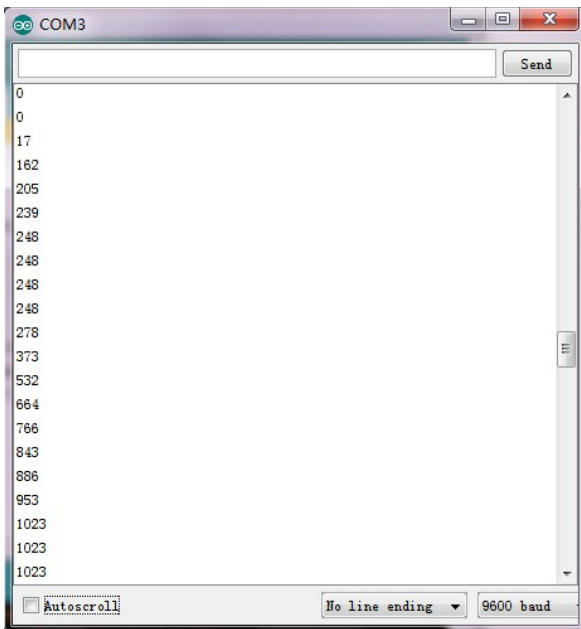
`delay(100); //delay in 0.1s`



}

## Test Result

Wire according to the figure above, burn the code and power on, the analog value is displayed on serial monitor, the date varies in the range of 0-1023 by rotating the potentiometer, as shown below:



## Project 7: 5V 1-channel Relay Module

### Description

In this course, we directly connect signal end of relay module to digital 3. The relay alternately turns on and off by controlling the digital 3.



## Equipment

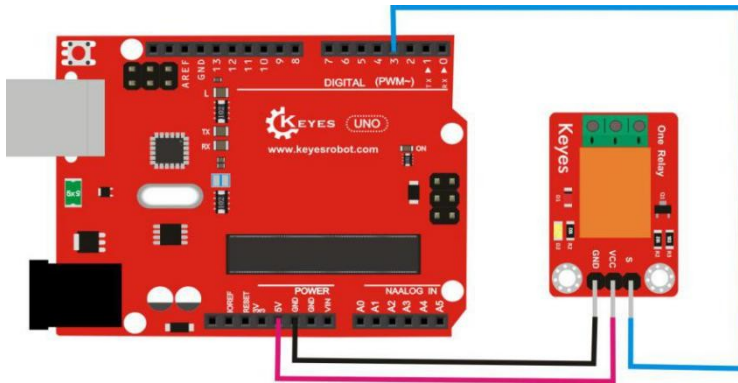
Development board \* 1

USB cable \* 1

5V single relay module \* 1

DuPont Lines

## Connection Diagram



## Test Code

```
int Relay = 3; //Define digital port3

void setup()
{
  pinMode(Relay, OUTPUT); //set Relay to OUTPUT}

void loop()
{
  digitalWrite(Relay, HIGH); //turn on relay
  delay(2000); //delay in 2s
```





```
digitalWrite(Relay, LOW); //turn off relay  
  
delay(2000); //delay in 2s  
  
}
```

## **Test Result**

Wire according to the figure above, upload the code and power on, the relay is turned on (ON terminal is connected, NC is disconnected) for 2s, the D2 light of relay is on, then stopped (ON terminal is disconnected, NC terminal is connected) for 2s, and alternately.

## **Project 8: Plug-in RGB Module**

### **Description**

This module is mainly made up of a plug-in full color LED. Full color mixing effect is produced by adjusting the basic color(red/blue/green) with PWM voltage input from R,G,B.

We can create cool and fancy light effect by controlling module. In this course, we will make RGB module alternately display different color.



## Equipment

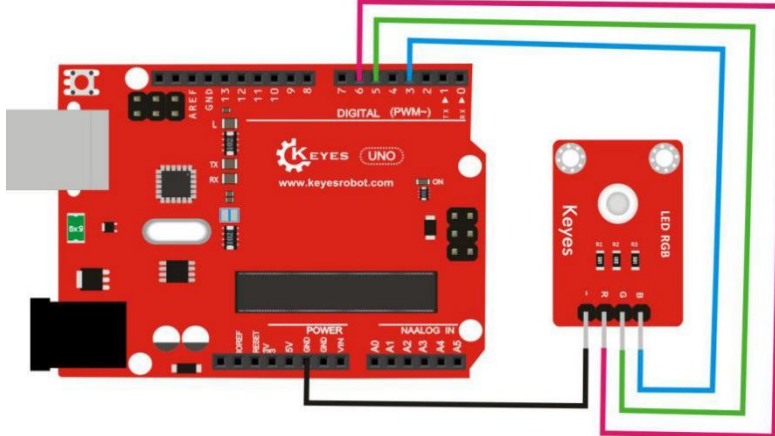
Development board \* 1

USB cable \* 1

Plug-in RGB module \* 1

DuPont Lines

## Connection Diagram



## Test Code

```
int redPin = 6; // R control pin of red LED is connected to pin 6 of Arduino
```

```
int greenPin = 5; // G control pin of green LED is connected to pin 5 of  
Arduino
```

```
int bluePin = 3; // B control pin of blue LED is connected to pin 3 of
```



## Arduino

```
void setup()
```

```
{
```

```
    pinMode(redPin, OUTPUT); //set redPin6 to output
```

```
    pinMode(greenPin, OUTPUT); //set greenPin5 to output
```

```
    pinMode(bluePin, OUTPUT); //set bluePin3 to output
```

```
}
```

```
void loop() // run over and over again
```

```
{
```

```
    // Basic colors:
```

```
    color(255, 0, 0); // red LED is on
```

```
    delay(1000); // delay in 1s
```

```
    color(0,255, 0); //green LED is on
```

```
    delay(1000); //delay in 1s
```

```
    color(0, 0, 255); // blue light is on
```

```
    delay(1000); //delay in 1s
```

```
    // Example blended colors:
```

```
    color(255,255,0); // yellow light is on
```

```
    delay(1000); //delay in 1s
```



```
    color(128,0,255); // Purple light is on
    delay(1000); //delay in 1s
    color(255,255,255); // white light is on
    delay(1000); //delay in 1s
    color(0,0,0); // turn off led
    delay(1000); //delay in 1s
}

void color (unsigned char red, unsigned char green, unsigned char
blue) //color controlling function
{
    analogWrite(redPin, red);
    analogWrite(greenPin,green);
    analogWrite(bluePin, blue);
}
```

### **Test Result**

After uploading the code and powering on, the RGB module will display red for 1s, green for 1s, blue for 1s, yellow for 1s, purple for 1s, white for 1s, stop for 1s, and alternate.



## **Project 9: Thermistors Sensor**

### **Description**

Based on the working principle of thermistors, it can sensor the temperature change of ambient environment. The data outputted by the sensor can be converted into the Celsius temperature value and displayed through programming, which is widely applied in gardening, home alarm systems and other devices. In the project, connect the signal end of sensor to A0 of development board, the temperature value in the current environment will be shown on serial monitor.

### **Equipment**

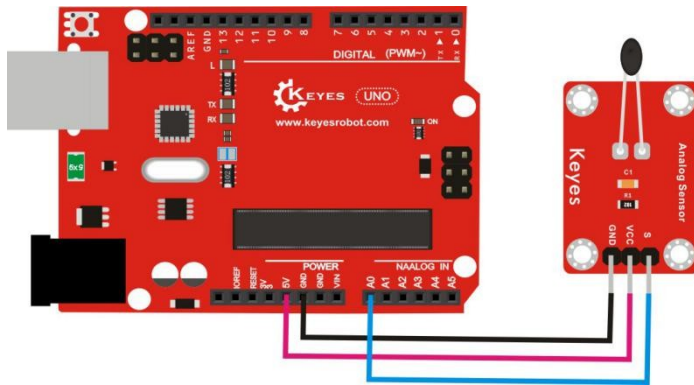
Development board \* 1

USB cable \* 1

Thermistor sensor \* 1

DuPont Lines

### **Connection Diagram**



## Test Code

```
#include <math.h>

double Thermister(int RawADC) {
double Temp;
Temp = log((((10240000/RawADC) - 10000));
Temp = 1 / (0.001129148 + (0.000234125 + (0.0000000876741 * Temp *
Temp ))* Temp );
Temp = Temp - 273.15; // Convert Kelvin to Celcius
return Temp;
}

void setup()
{
Serial.begin(9600); //set baud rate
}

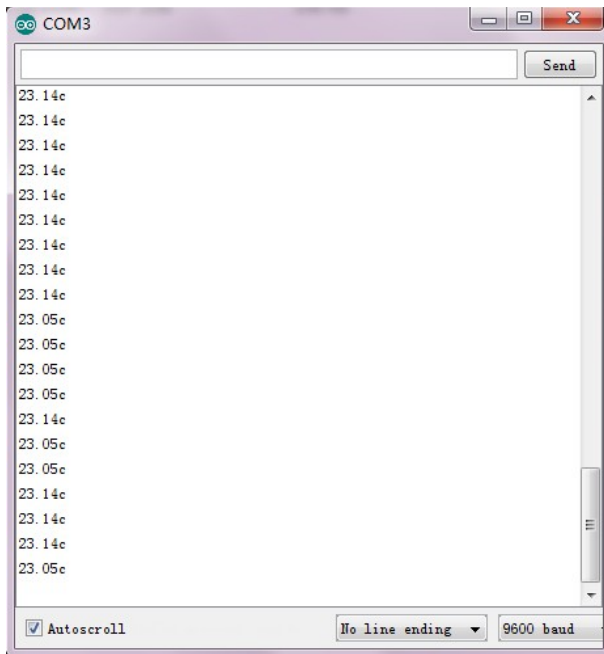
void loop()
{
Serial.print(Thermister(analogRead(0))); // display temperature value
```



```
Serial.println("c"); // show c, and wrap word  
  
delay(500); // delay in 0.5S  
  
}
```

## Test Result

Wire according to the above figure, upload the code, and after power on, we can see the current temperature value on the serial monitor, as shown below.



## Project 10: Button Sensor

### Description

When the button is pressed, the low-level is outputted at the sensor signal terminal,; and when released, the sensor signal terminal remains high level.



In the project, we control the D13 indicator on and off via sensor.

## Equipment

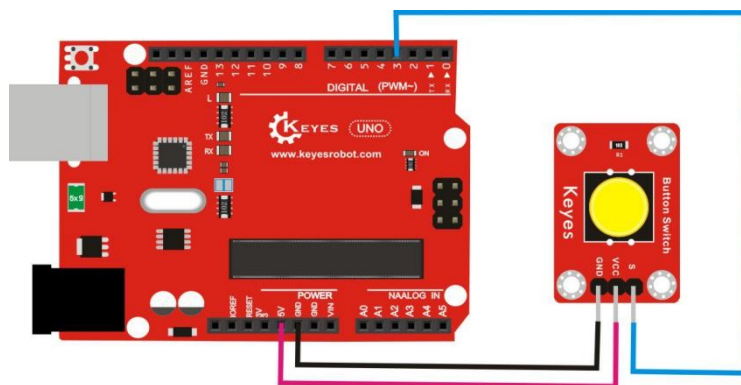
Development board \* 1

USB cable \* 1

Button sensor \* 1

DuPont Lines

## Connection Diagram



## Test Code

```
int ledPin = 13; //Define digital port13
```

```
int inputPin = 3; //Define digital port3
```

```
void setup()
```





```
{  
  
pinMode(ledPin, OUTPUT); //set ledPin to OUTPUT  
pinMode(inputPin, INPUT); //set inputPin to INPUT  
  
}  
  
void loop()  
  
{  
  
int val = digitalRead(inputPin);  
  
//Set the digital variable val, read the value of digital port 3, and assign it to  
val  
  
if (val == LOW) //Low level at val, LED lights up  
{  
  
digitalWrite(ledPin, HIGH); // LED lights on  
  
}  
  
else  
  
{  
  
digitalWrite(ledPin, LOW); // LED gets dark  
  
}  
  
}
```

### **Test Result**

Wire according to the figure above, upload the code and power on, the D13 indicator is on when the button is pressed. The D13 indicator goes out when released.



## **Project 11: DHT11 Temperature and Humidity Sensor**

### **Description**

It is an integrated temperature and humidity sensor with calibrated digital signal output, applied to digital module acquisition and temperature and humidity sensing technology. Therefore, the extremely high reliability and excellent long-term stability can be kept.

In the project, connect signal end of sensor to the digital 3 of development board, the present temperature and humidity values can be displayed on serial monitor.

.

### **Equipment**

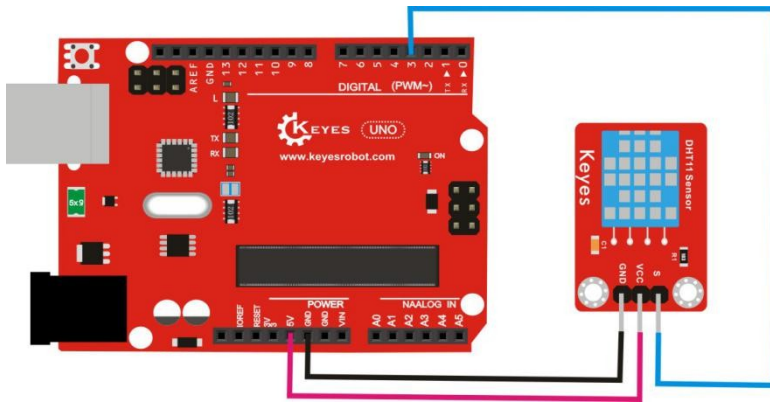
Development board \* 1

USB cable \* 1

DHT11 temperature and humidity sensor \* 1

DuPont Lines

### **Connection Diagram**



## Test Code

```
#include <dht11.h>

dht11 DHT;

#define DHT11_PIN 3

void setup(){

  Serial.begin(9600);

  Serial.println("DHT TEST PROGRAM ");

  Serial.print("LIBRARY VERSION: ");

  Serial.println(DHT11LIB_VERSION);

  Serial.println();

  Serial.println("Type,\tstatus,\tHumidity (%),\tTemperature (C)");

}

void loop(){

  int chk;

  Serial.print("DHT11, \t");

  chk = DHT.read(DHT11_PIN);  // READ DATA

  switch (chk){
```



```
case DHTLIB_OK:
    Serial.print("OK,\t");
    break;
case DHTLIB_ERROR_CHECKSUM:
    Serial.print("Checksum error,\t");
    break;
case DHTLIB_ERROR_TIMEOUT:
    Serial.print("Time out error,\t");
    break;
default:
    Serial.print("Unknown error,\t");
    break;
}

// DISPLAT DATA
Serial.print(DHT.humidity,1);
Serial.print(",\t");
Serial.println(DHT.temperature,1);
delay(1000);
}
```

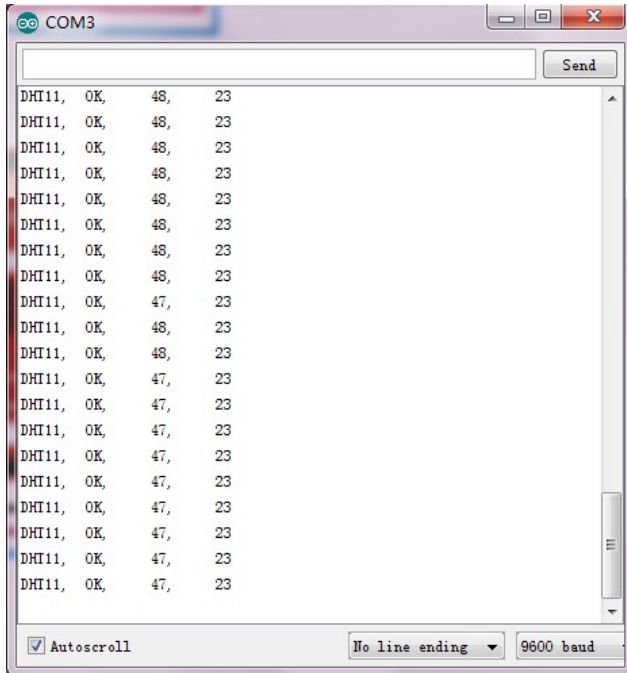
Download link for library file

<https://pan.baidu.com/s/1eSIMYD8>



## Test Result

Wire via the above figure, upload the code and power on, the present temperature and humidity values can be displayed on serial monitor.



## Project 12: Photocell Sensor

### Description

It is used to detect ambient light

Sensitive to ambient light, this sensor is applied to detect the ambient.

Then the corresponding value is shown and triggers the MCU and relay module.

In this project, connect the signal of the sensor to the analog port A0 of the



development board. The corresponding analog value is shown on the serial monitor.

## Equipment

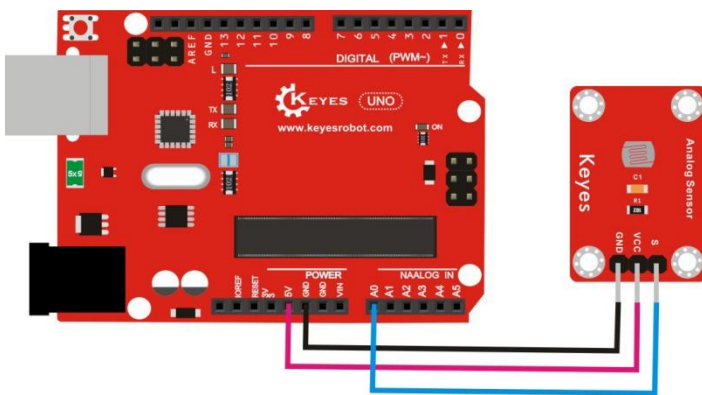
Development board \* 1

USB cable \* 1

Photoresistor sensor \* 1

DuPont Lines

## Connection Diagram



## Test Code

```
int sensorPin =A0 ; //Define analog port A0
```

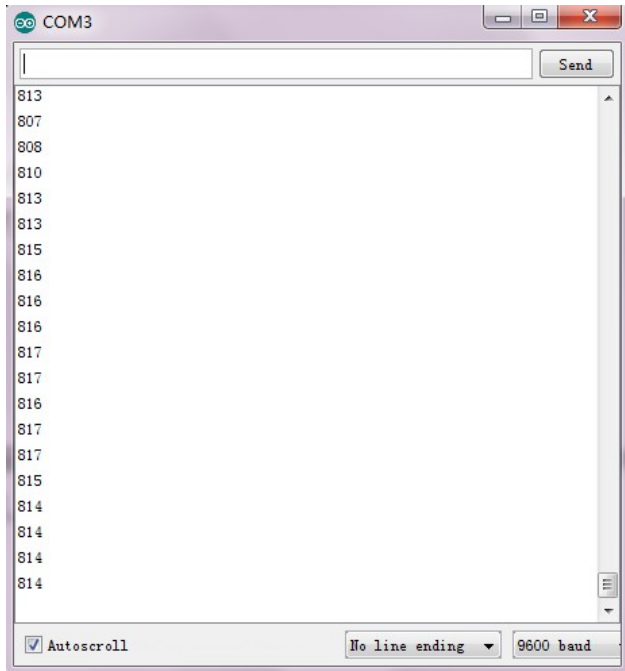
```
int value = 0; //set value to 0
```



```
void setup()
{
  Serial.begin(9600); //set baud rate
}
void loop()
{
  value = analogRead(sensorPin); //set value as reading the value of A0
  Serial.println(value, DEC); //show value of "value", wrap word
  delay(200); //delay in 0.2s
}
```

### **Test Result**

Wire according to the above figure, upload the code and power on, we will find that the analog value is displayed on the serial monitor. The more brighter the light is, the larger the value is. As shown below:



## Project 13: Tilt Sensor

### Description

Tilt sensor is a digital tilt switch. It can be used as a simple tilt sensor. Tilt sensors (tilt ball switch) allow you to detect orientation or inclination. They are small, inexpensive, low-power and easy-to-use.

Simply connect the sensor to our IO/sensor shield, you can make amazing interactive projects. In the project, we control the brightness of D13 via it.

### Equipment

Development board \* 1



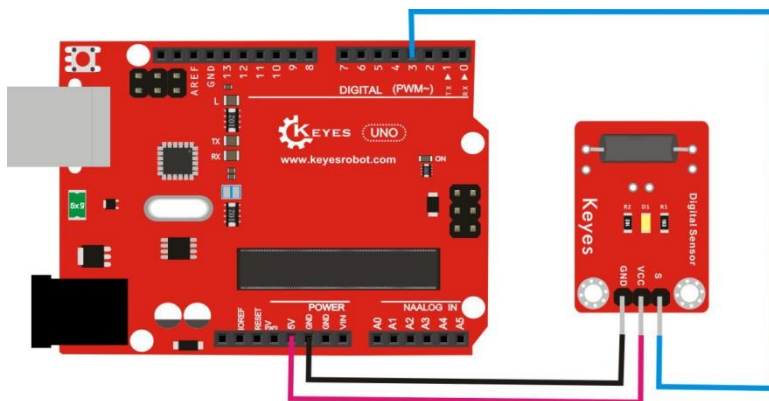


USB cable \* 1

Tilt sensor \* 1

DuPont Lines

## Connection Diagram



## Test Code

```
int ledPin = 13; //Define digital port13
int switcher = 3; // Define digital port3
void setup()
{
  pinMode(ledPin, OUTPUT); // set ledPin to output
  pinMode(switcher, INPUT); //set switcher to input
}
void loop()
{
  if(digitalRead(switcher)==HIGH) //High level is at digital 3
```



```
{  
digitalWrite(ledPin, HIGH); // LED lights on}  
else  
{  
digitalWrite(ledPin, LOW); // LED gets dark  
}  
}
```

### **Test Result**

Upload the code to the board and power on, then tilt the sensor, you will see D13 is turned on. When tilt in opposite direction, the D13 is turned off.

## **Project 14: Microphone Sound Sensor**

### **Description**

This sensor is mainly used to test sound.

The S pin of this sensor is analog output, that is voltage signal real-time output of microphone. In the project, the S end is connected to A0 of development board, and the analog value can be read on serial monitor.



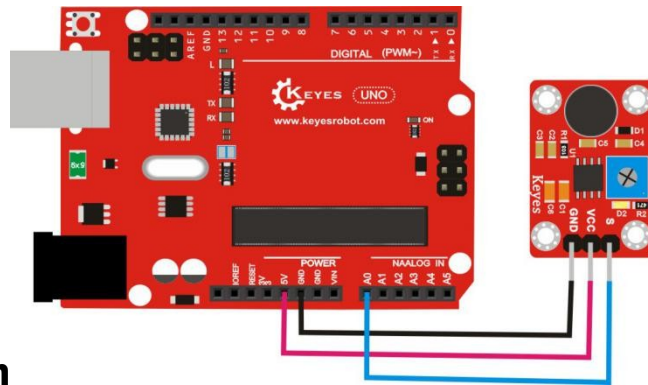
## Equipment

Development board \* 1

USB cable \* 1

Microphone sound sensor \* 1

DuPont Lines



## Connection Diagram

## Test Code

```
int sensorPin =A0 ; //Define analog portA0

int value = 0; //set value to 0

void setup()

{

  Serial.begin(9600); //set baud rate

}

void loop()

{

  value = analogRead(sensorPin); //set value to reading the value of A0

  Serial.println(value, DEC); //display value, and wrap word

  delay(100); //delay in 0.1s
```





This is a hall magnetic sensor. It excels in small size, high sensitivity, response speed, temperature performance, as well as high reliability. Therefore, it is widely applied to non-contact switch, position and speed detection and control, alarm device, textile control system, etc. We control D13 of Arduino UNO board on and off via sensor.

## Equipment

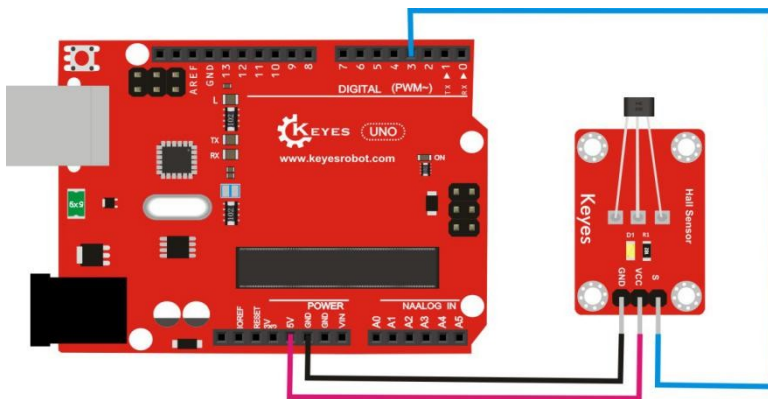
Development board \* 1

USB cable \* 1

Hall Magnetic Sensor\* 1

DuPont Lines

## Connection Diagram



## Test Code

```
int ledPin = 13; //Define digital port13
```



```
int inputPin = 3; //Define digital port3

int val = 0; // Define the digital variable val and set to 0

void setup()

{

pinMode(ledPin, OUTPUT); //set ledPin to output

pinMode(inputPin, INPUT); //set inputPin to input

}

void loop(){

val = digitalRead(inputPin); //read the value of digital 3, assign the value to

val

if (val == LOW) //when low level at val, LED lights on

{

digitalWrite(ledPin, HIGH); //LED lights on

}

else

{

digitalWrite(ledPin, LOW); //LED gets dark

}

}
```



## **Test Result**

Wire according to the above figure, upload the code, and after power on, D13 of Arduino UNO board is off, as well as D1 of the sensor. Whereas the magnet is approaching to the sensor, the D13 and D1 are turned on.

## **Project 16: Crash Sensor**

### **Description**

When the button is pressed due to crashing object , the sensor signal terminal outputs the low level, and when the button is released, the sensor signal end maintains a high level. The sensor can be used as a limit switch in a 3D printer. In the project, we control D13 of Arduino UNO board on and off via sensor.

### **Equipment**

Development board \* 1

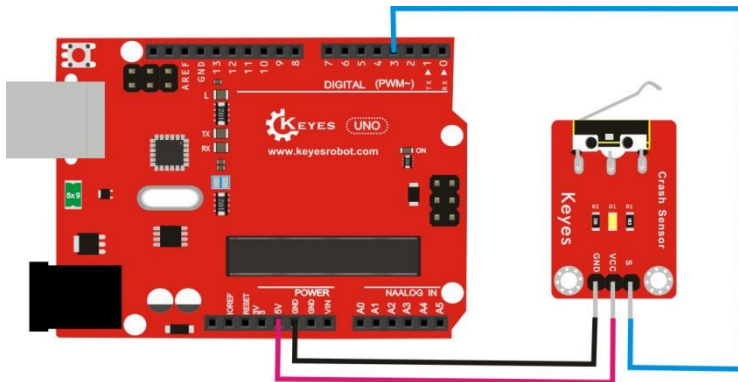
USB cable \* 1



Crash sensor \* 1

DuPont Lines

### Connection Diagram



### Test Code

```
int Led=13;//Define LED interface
int Shock=3;//Define the interface of crash sensor
int val;//Define digital variables val
void setup()
{
pinMode(Led,OUTPUT);//set LED to output interface
pinMode(Shock,INPUT);//Define crash sensor as the output interface}
void loop()
{
val=digitalRead(Shock);//read the value of digital 3, assign it to val
if(val==LOW)//when crash sensor detects the signal, LED lights on
{
```





```
digitalWrite(Led,HIGH); //LED lights on
}
else
{
digitalWrite(Led,LOW); //LED gets dark
}
}
```

### **Test Result**

Wire according to the above figure, burn the code and power on, press down the small iron sheet of sensor, the D13 indicator on the Arduino UNO board and the D1 on the sensor light up, otherwise, the D1 and D13 are off.

## **Project 17: Knock Sensor**

### **Description**

Mainly composed of SW-280 vibration switch, knock sensor is an inductive proximity switch due to pass sensing result to circuit device by sensing how strong the vibration is.



In the project, we control the D13 light of Arduino UNO board via sensor

## Equipment

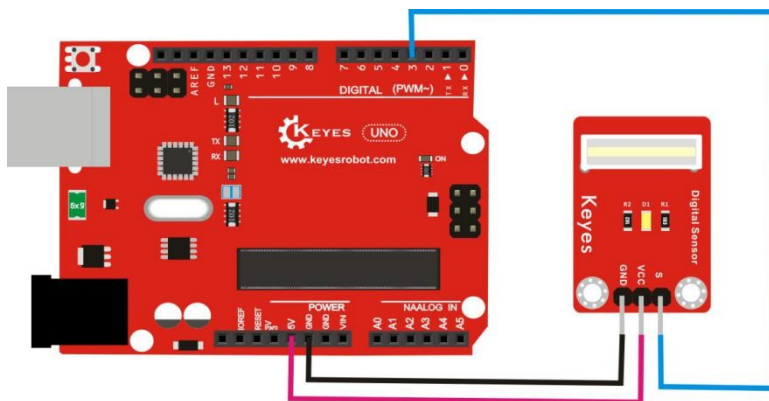
Development board \* 1

USB cable \* 1

Knock module sensor \* 1

DuPont Lines

## Connection Diagram



## Test Code

```
int Led=13;//Define digital port13
int Shock=3;//Define digital port3
int val;//Define digital variables val
void setup()
{
pinMode(Led,OUTPUT);//set Led to output
pinMode(Shock,INPUT);//set Shock to input
```



```
}  
  
void loop()  
{  
  val=digitalRead(Shock); //Read the value of digital port 3 and assign it to  
  val  
  if(val==LOW)      //Led lights up when val is low level  
  {  
    digitalWrite(Led,HIGH); //Led lights on  
  }  
  else  
  {  
    digitalWrite(Led,LOW); //Led gets dark  
  }  
}
```

### **Test Result**

Wire according to the figure above, upload the code, and after power on, tap the module. The D13 LED of the Arduino UNO board and the D1 light of this sensor are on.



## Project 18: Obstacle Avoidance Sensor

### Description

After power on, the signal terminal S outputs 0 when the sensor detects the object, on the contrast, 1 is outputted at the signal end. The inductive sensitivity can be adjusted by rotating the potentiometer. It is applicable to obstacle avoidance cars, line tracking, anti-falling, counter, assembly line cutting, liquid level detection, etc.

In the project, we control the D13 light of Arduino UNO board via sensor

### Equipment

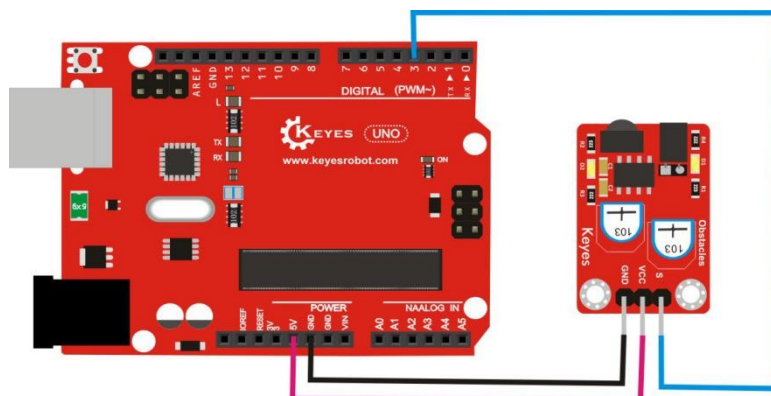
Development board \* 1

USB cable \* 1

Obstacle avoidance sensor \* 1

DuPont Lines

### Connection Diagram





## Test Code

```
const int sensorPin = 3; //Define digital port13

const int ledPin = 13; //Define digital port3

int sensorState = 0; //Define digital variables sensorState, set to 0

void setup()

{

pinMode(ledPin, OUTPUT); //set ledPin to output

pinMode(sensorPin, INPUT); //set sensorPin to input

}

void loop()

{

sensorState = digitalRead(sensorPin);

//Read the value of digital port 3 and assign it to sensorState

if (sensorState == LOW) //LED lights up when sensorState is low

{

digitalWrite(ledPin, HIGH); //LED lights on

}

else

{

digitalWrite(ledPin, LOW); //LED gets dark

}

}
```



## **Test Result**

Wire according to the figure above, upload the code, and after power on. Observe the D1 when adjusting the potentiometer. The sensing distance is longest when the critical point is maintained.

When the obstacle is around the sensor, the D1 light of the sensor and D13 indicator are turned off. When no obstacle is around, the D1 and D13 lights are on.

## **Project 19: LM35 Temperature Sensor**

### **Description**

LM35 linear temperature sensor is based on semiconductor LM35 temperature sensor. It can be used to detect ambient air temperature.

This sensor offers a functional range among 0 degree Celsius to 100 degree Celsius. Sensitivity is 10mV per degree Celsius. The output voltage is proportional to the temperature.

In this project, connect the signal of the sensor to the analog port A0 of the development board. You can see the current temperature value on the serial monitor.



## Equipment

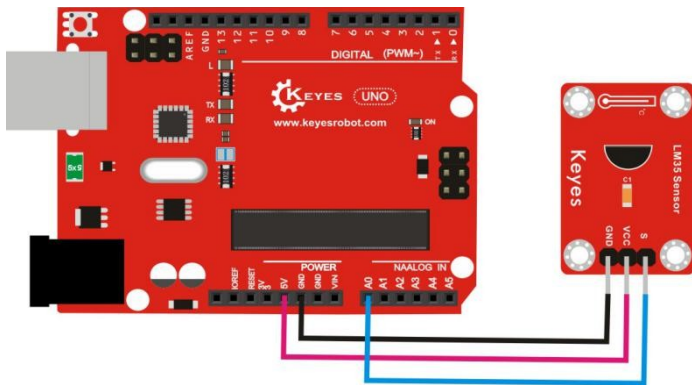
Development board \* 1

USB cable \* 1

LM35 temperature sensor \* 1

Dupont Line

## Connection Diagram



## Test Code

```
void setup()
```

```
{
```

```
Serial.begin(9600); //set baud rate
```

```
}
```

```
void loop()
```

```
{
```

```
int val; //Define digital variablesval
```

```
int dat; //Define digital variablesdat
```

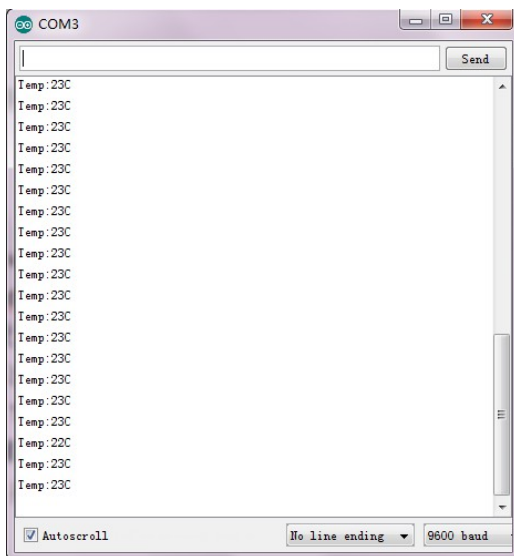
```
val=analogRead(0); //set val to reading the value of A0
```



```
dat=(500 * val) /1024; //Calculate the current temperature dat  
Serial.print("Temp:"); //display Temp:  
Serial.print(dat); //Display the temperature value  
Serial.println("C");//dispaly C, and wrap word  
delay(500); //delay in 0.5S  
}
```

## Test Result

Wire according to the figure above, upload the code, and after power on. The current temperature value can be displayed on the serial monitor, as shown below.







## Project 20: Laser Head Sensor

### Description

Made up of light emitting die, condenser lens and copper adjustable sleeve, it can be applied to laser toys, electronic pointer, electronic spirit level, miniature LCD projection.

### Equipment

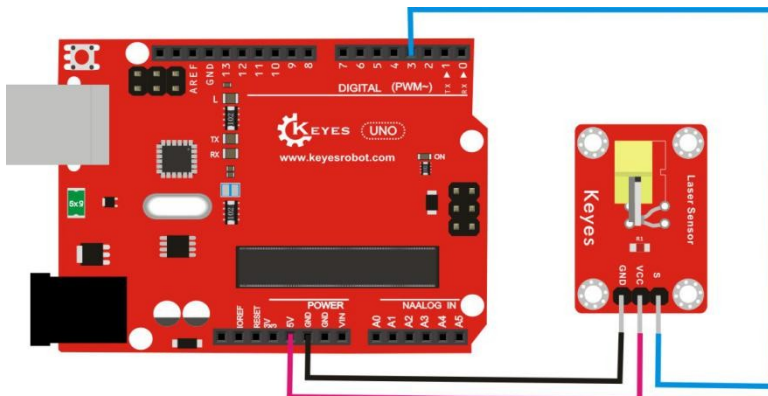
Development board \* 1

USB cable \* 1

Laser head sensor \* 1

DuPont Lines

### Connection Diagram



### Test Code



```
void setup()
{
  pinMode(3, OUTPUT); //Define pin 3 as a digital output interface
}

void loop() {
  digitalWrite(3, HIGH); // turn on laser head
  delay(1000); // delay in 1s
  digitalWrite(3, LOW); // turn off laser head
  delay(1000); // delay in 1s
}
```

### **Test Result**

Wire according to the figure above, upload the code and power on, the laser head is turned on for 1s and turned off for 1s, and alternately.

## **Project 21: Line Tracking Sensor**

### **Description**

The working principle of the TCRT5000 infrared tube on the sensor is to convert the reflected signal into a current signal by reacting to reflectivity



differently. Signal 1 is outputted and circuit is off when detecting black color, signal 0 is outputted and circuit is on when detecting white color. The detection height is 0-3cm.

In the circuit you can adjust the sensitivity by rotating potentiometer.

Connect the S terminal of the sensor to the digital port D3 of the development board, and you can see the corresponding digital outputted value on the serial monitor.

## Equipment

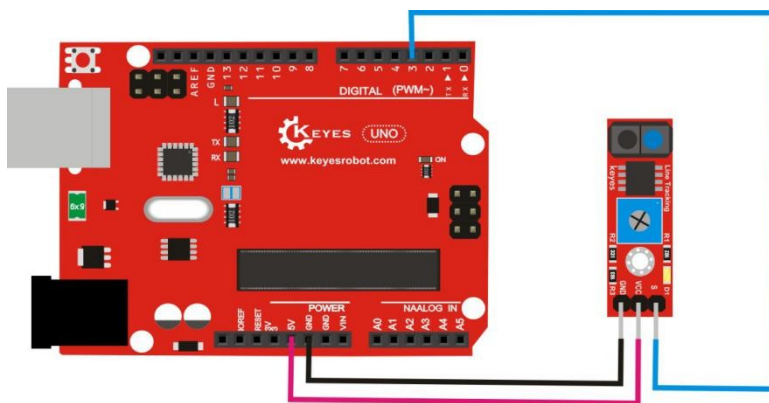
Development board \* 1

USB cable \* 1

Line tracking sensor \* 1

DuPont Lines

## Connection Diagram



## Test Code

```
void setup()
```

```
{
```

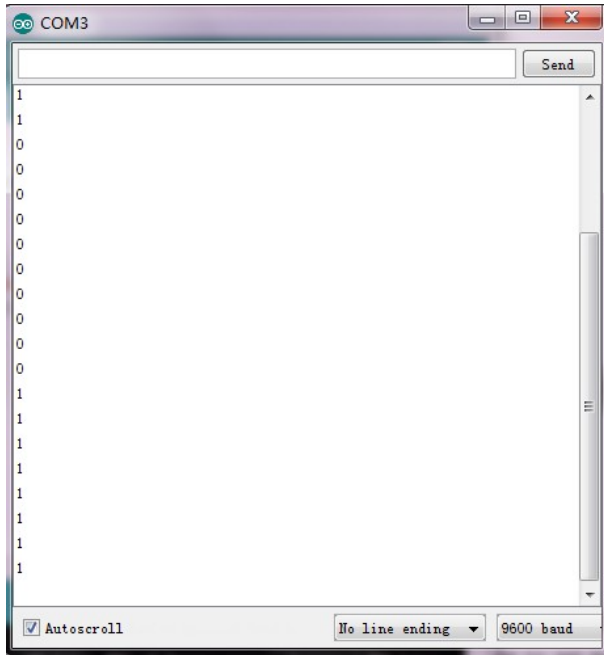


```
Serial.begin(9600); //set baud rate
}
void loop()
{
  Serial.println(digitalRead(3)); // Output the value read from digital port 3,
  and wrap word
  delay(500); //delay in 0.5s
}
```

### **Test Result**

Wire according to the figure above, upload the code, and after power on, high level is outputted, 1 is displayed on serial monitor and D1 of sensor is off when detecting the black object; the signal end outputs low level, 0 is displayed on serial monitor, D1 is on.

Rotating the potentiometer can adjust the sensitivity. The sensitivity is the highest, when D1 is adjusted to the critical point of on and off.



## Project 22: 18B20 Temperature Sensor

### Description

This sensor excels in small size, strong anti-interference ability and high accuracy. The temperature range is  $-55 \sim +125$  °C, inherent temperature resolution 1 °C. In the project, we connect the signal end of sensor to digital 3 of development board, the current temperature value will be displayed on the serial monitor.

### Equipment

Development board \* 1

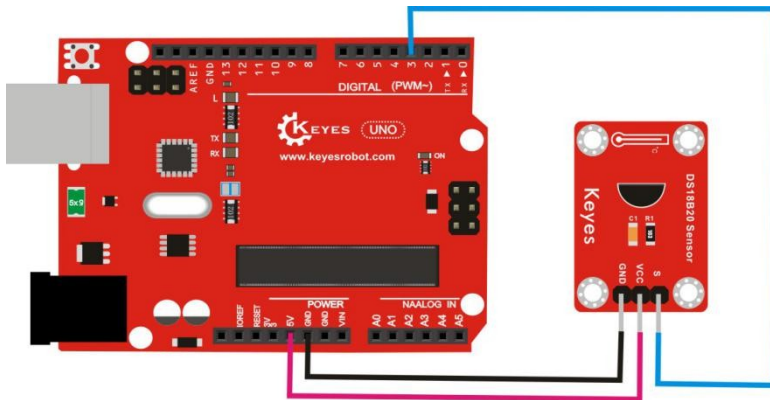


USB cable \* 1

18B20 temperature sensor \* 1

DuPont Lines

### Connection Diagram



### Test Code

```
#include <OneWire.h>

int DS18S20_Pin = 3; //Define digital port3

OneWire ds(DS18S20_Pin);

void setup(void) {
  Serial.begin(9600); //set baud rate
}

void loop(void) {
  float temperature = getTemp(); //Calculate temperature value via function
  Serial.println(temperature); //Display temperature value, and wrap word
  delay(100); //delay in 0.1S
}
```



```
float getTemp(){
    //returns the temperature from one DS18S20 in DEG Celsius

    byte data[12];
    byte addr[8];
    if ( !ds.search(addr)) {
        //no more sensors on chain, reset search
        ds.reset_search();
        return -1000;
    }
    if ( OneWire::crc8( addr, 7) != addr[7]) {
        Serial.println("CRC is not valid!");
        return -1000;
    }
    if ( addr[0] != 0x10 && addr[0] != 0x28) {
        Serial.print("Device is not recognized");
        return -1000;
    }
    ds.reset();
    ds.select(addr);
    ds.write(0x44,1); // start conversion, with parasite power on at the end
    byte present = ds.reset();
    ds.select(addr);
```



```
ds.write(0xBE); // Read Scratchpad

for (int i = 0; i < 9; i++) { // we need 9 bytes
  data[i] = ds.read();
}

ds.reset_search();

byte MSB = data[1];
byte LSB = data[0];

float tempRead = ((MSB << 8) | LSB); //using two's compliment
float TemperatureSum = tempRead / 16;

return TemperatureSum;
}
```

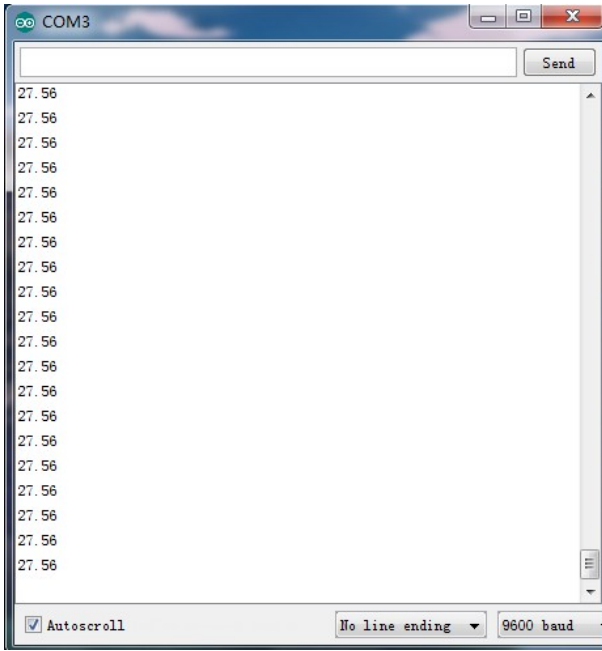
### **Download link for library file**

<https://pan.baidu.com/s/1o7HKVKQ>

### **Test Result**

Wire according to the figure above, upload the code and power-on. We can see the current ambient temperature value on the serial monitor, as shown below.





## 5. Resource

<https://pan.baidu.com/s/1sIFTsQD>