Ultrasonic Intelligent Obstacle Avoidance Vehicle Manual

1. Introduction

ARDUINO ultrasonic intelligent obstacle avoidance vehicle is a microcontroller learning application development system based on the arduino microcontroller series
Atmega-328 is used as the core to complete the function of ultrasonic obstacle avoidance. The suite contains a large number of interesting programs and can be expanded with external
circuit module, thereby increasing the functionality of the car. It is designed to allow users to escape boredom when learning ARDUINO microcontrollers.
theoretical knowledge and the ability to develop microcontroller systems while having fun.

2. Parameter

1. Motor parameters: voltage range: 1.5-12V, motor shaft length: 10mm, speed 6.0V 100rpm/min.
2. The L298N drive module is used to control the motor, which is truly isolated from the microcontroller.
3. The obstacle avoidance part uses HC-SR04 ultrasonic, which has stable performance and accurate distance measurement.
6. Can be connected to external voltage of 7~12V. It can be equipped with a variety of sensor modules to realize various functions according to your imagination.

3. Introduction to experimental courses

1. Application of L298N motor driver board
2. Application of ultrasonic module
3. Ultrasonic obstacle avoidance smart car

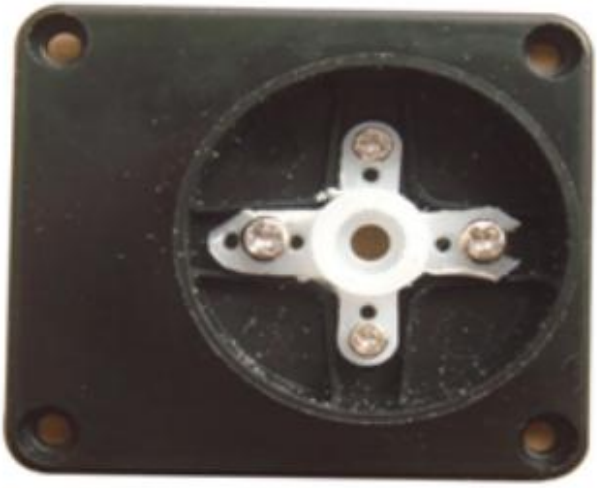
4. Ultrasonic obstacle avoidance vehicle gimbal installation instructions

Gimbal steering gear ultrasonic installation diagram

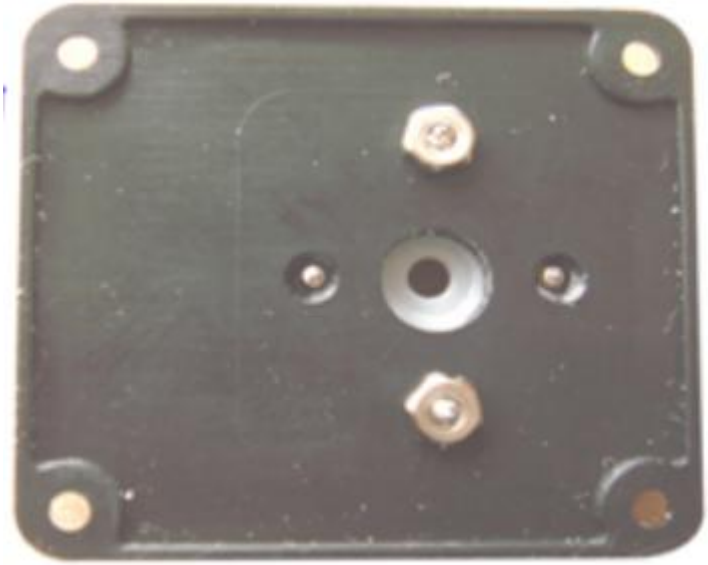
Take out the cross colloid from the steering gear accessories bag

Cut the cross into equal lengths on all four sides and polish it to the same width.

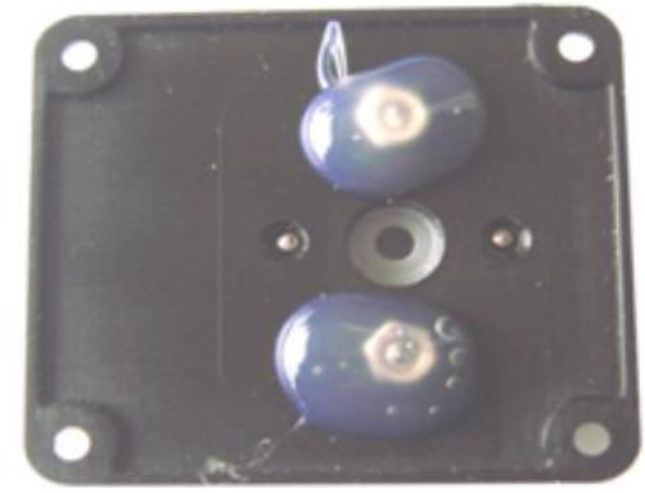
As shown in the picture, install the 2*8mm and 1.2*5mm screws to the second hole of the cross to the gimbal base.



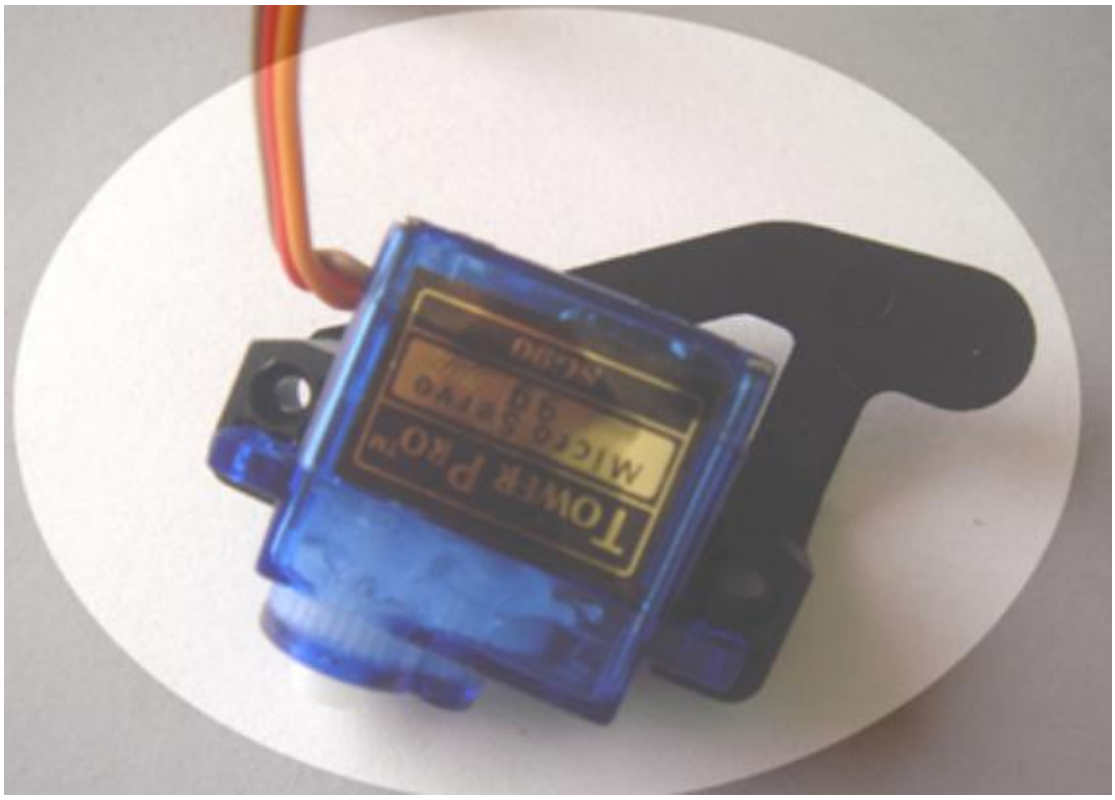
Put the nut on the 2*8mm screw position at the bottom of the gimbal



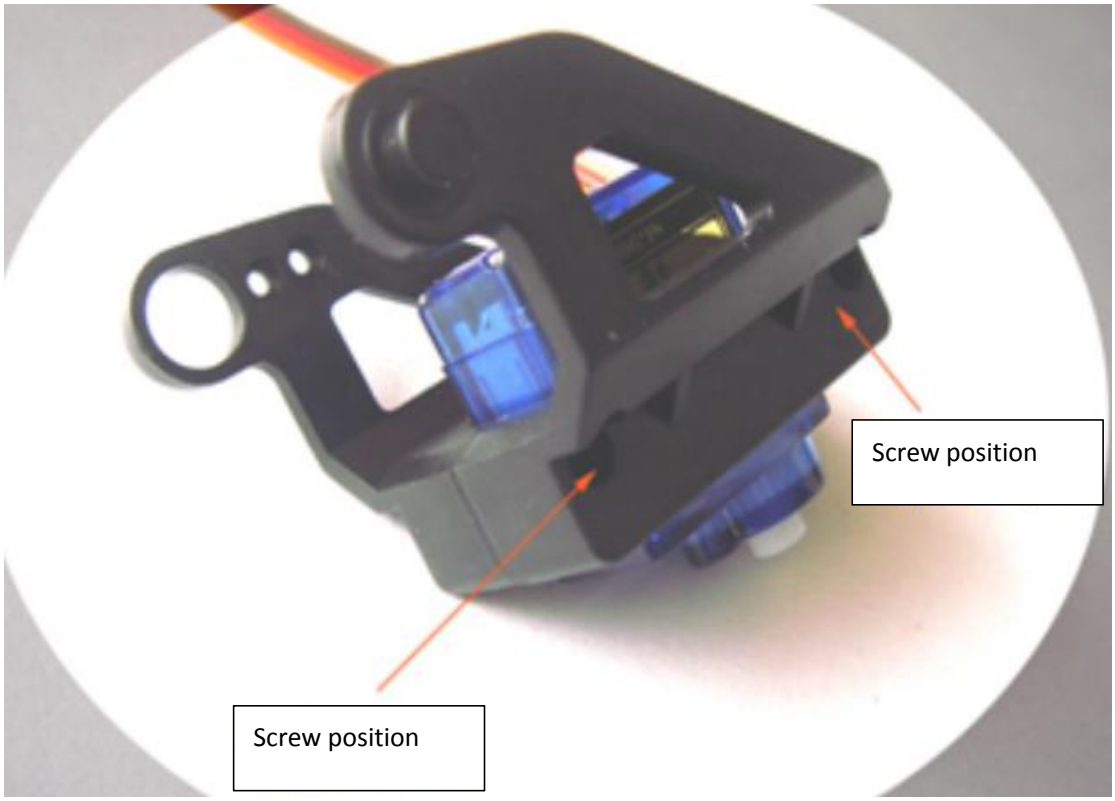
Apply hot melt glue to fix the screws



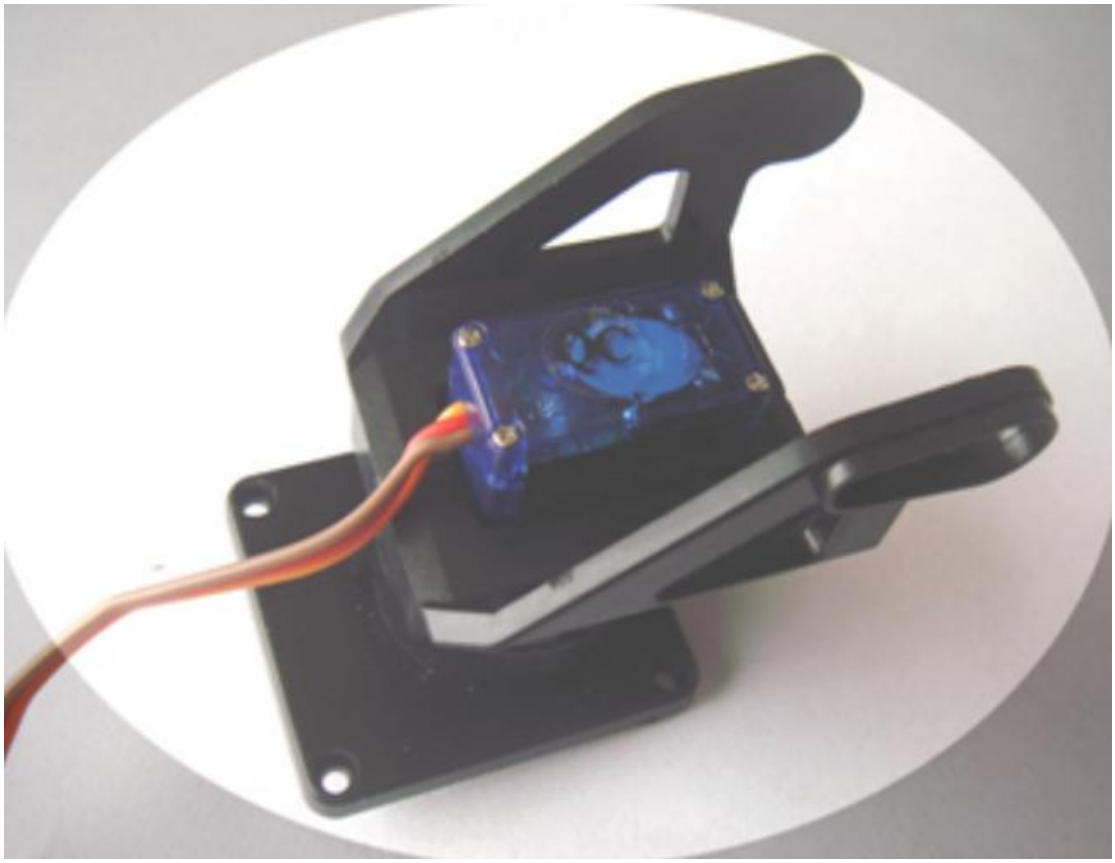
Install the servo on the two edges of the gimbal



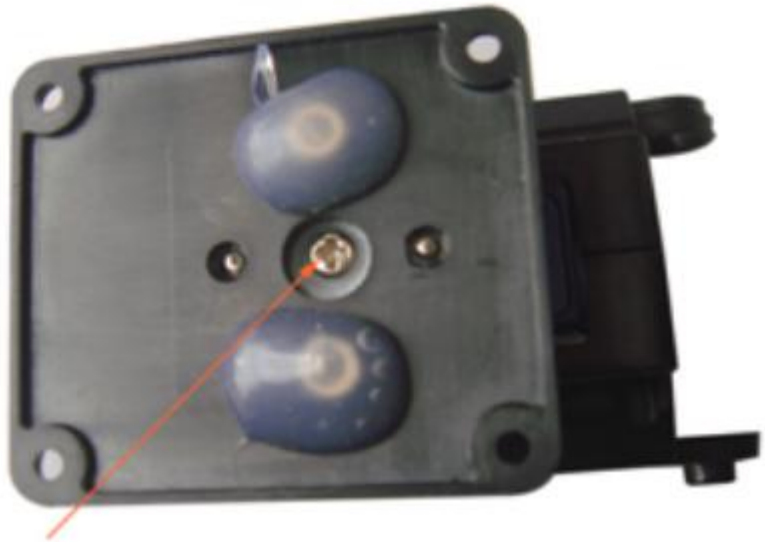
After installation, secure it with screws



Put the installed servo into the fixed cross colloid and adjust the direction



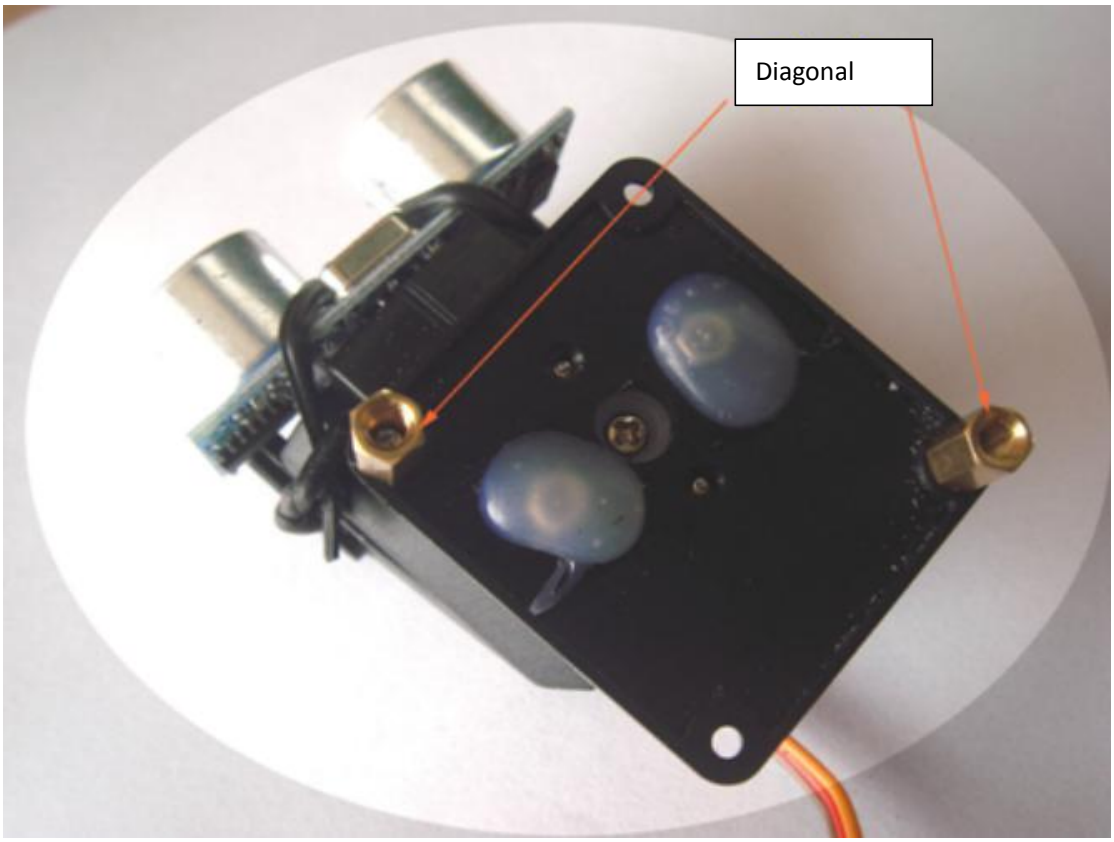
Take out the 2*6mm screws from the servo component package and install them into the servo fixing holes.

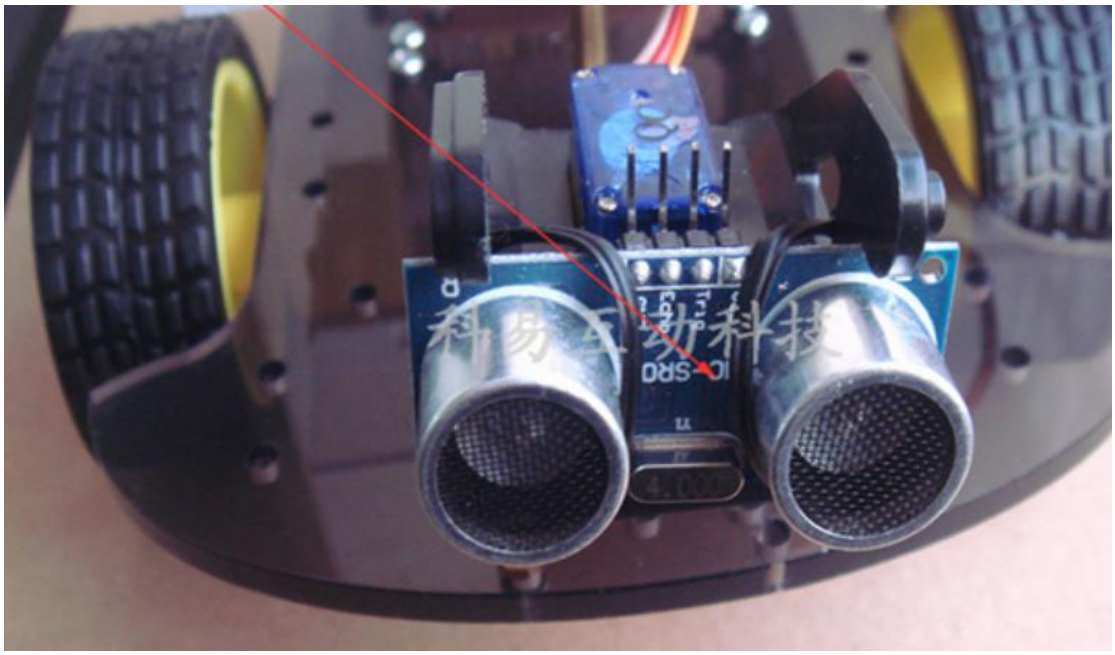


Servo fixing hole

Use a tie to secure the ultrasonic module to the front of the gimbal

Install the 6mm copper pillar into the mounting hole of the gimbal base

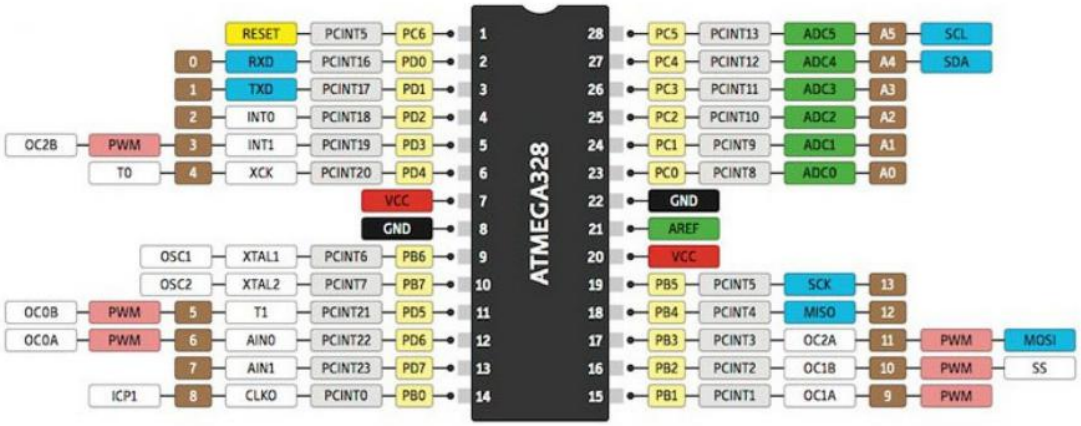




5. Use of Arduino microcontroller

1.Introduce:

Arduino is an open source hardware project platform originating from Italy. The platform includes a circuit board with simple I/O functions and a set of program development environment software. Arduino can be used to develop interactive products. For example, it can read a large number of switch and sensor signals, and can control lights, motors, and other various physical devices; Arduino can also develop peripheral devices connected to PCs that can be used on The runtime communicates with the software on the PC. Arduino's hardware circuit board can be soldered and assembled by yourself, or you can purchase already assembled modules, and the software of the program development environment can be downloaded and used for free from the Internet.



Let’s see how the Arduino team defines it:

Arduino is an open source electronics prototyping platform with flexible, easy-to-use hardware and software. Arduino is designed for designers, arts and crafts people, hobbyists, and anyone interested in developing interactive installations or interactive development environments.
Arduino can receive input signals from various sensors to detect the operating environment and affect its surroundings by controlling light sources, motors, and other actuators. The microcontroller on the board is programmed using the Arduino programming language (based on Wiring) and the Arduino development environment (based on Processing). Arduino can run independently or communicate with software running on your computer (e.g., Flash, Processing, MaxMSP).
The Arduino hardware circuit board can be soldered and assembled by yourself, or you can buy it already assembled. The software can be downloaded and used for free from the Arduino website. You can obtain hardware reference designs (CAD files) under an open source license and freely modify them to suit your needs.
The definition of Arduino is still a bit fuzzy, and this is where Arduino excels. Arduino is the glue that connects people's various tasks. To give the most accurate definition of Arduino, it is best to describe it with some examples.
Do you want your coffee pot to squeak to remind you when your coffee is ready?
Do you want your phone to send out an alert to notify you when there is a new email in your mailbox?
Want a sparkly stuffed animal?
Want a Professor X steampunk-style wheelchair with voice and drink delivery?
Want a set of shortcut keys you can experiment with to test your buzzer?
Want to make a homemade Metroid arm cannon for your son?
Want to make your own heart rate monitor and store the records of each cycling session into a memory card?
Have you ever thought about making your own robot that can draw on the ground and ride in the snow?

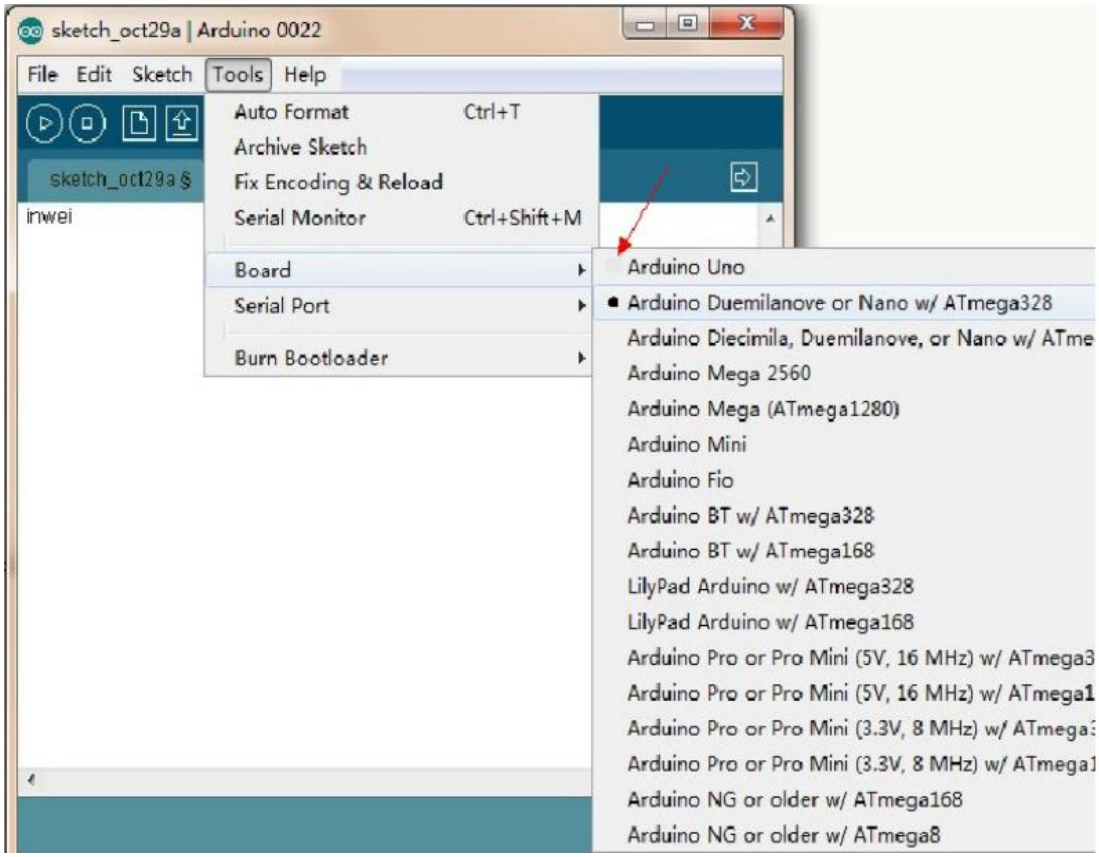
2. Arduino driver installation and program programming

First download the Arduino development software, web address:
http://arduino.cc/en/Main/Software
The downloaded file is a compressed folder of arduino-0023.zip, unzip it to the hard drive.
When the Arduino power board is connected to Windows via a USB cable, it will prompt that a new USB device named "FT232R USB UART" has been found. Then Windows will guide us to the "Found New Hardware Wizard" window, select "No, temporarily" Click the "Next" button after selecting the "No" option:

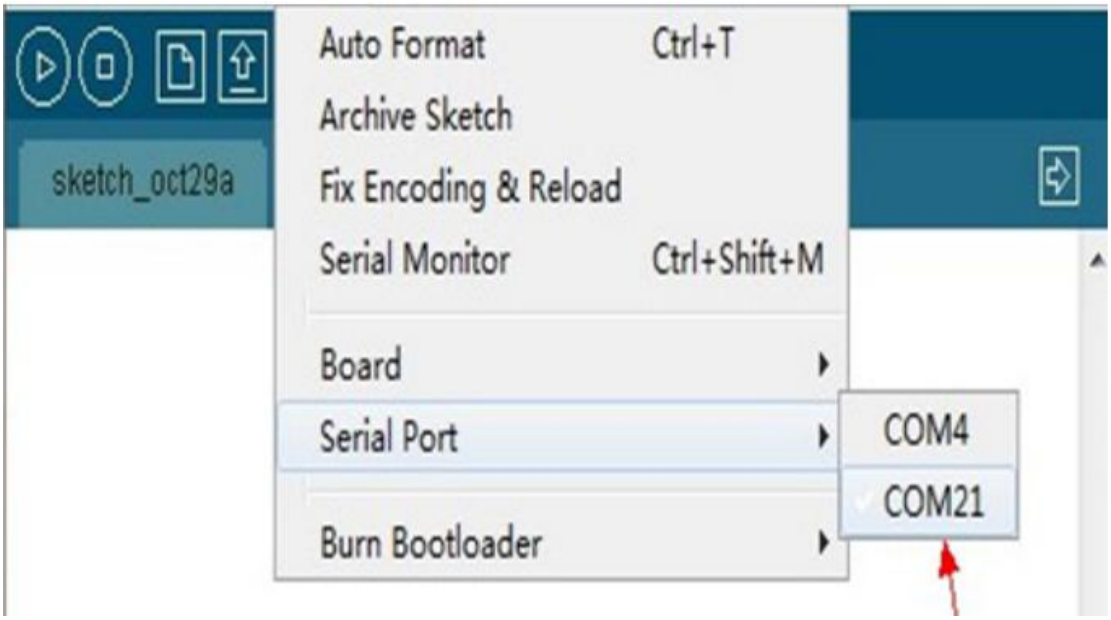
The next step is to install the drivers required by Arduino. Select the "Install from list or specified location (advanced)" option and click the "Next" button:
Arduino's USB driver is placed in the drivers directory under the Arduino 0021 installation directory. We need to indicate to Windows that this directory is the directory to be searched when installing the driver:

After the Arduino USB driver is successfully installed, we can find the corresponding Arduino serial port in the Windows Device Manager.

The following demonstrates the programming of the first program, lighting the "L" light
In the programming interface of Arduino-0023, click [Tools], move the mouse to the [Board] option of the drop-down menu, and look in the submenu that continues to pop up to see if there is a black dot in front of [arduino Duemilanove]. If not, click Click on the [arduino Duemilanove] option.

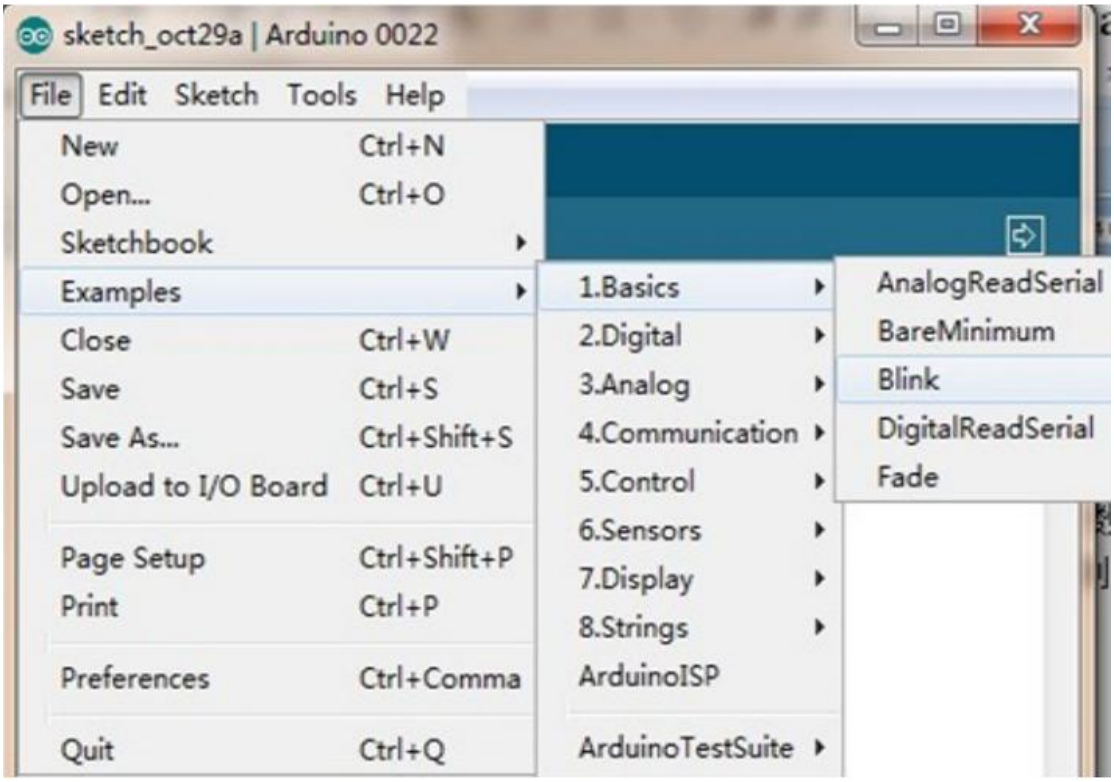


The next step is to select the correct COM connection. Do you still remember the value of X (COMX) that you were asked to record when you just installed the hardware? It will be used here. For example, the port of the Arduino just installed is 21, so click 21 with the mouse.

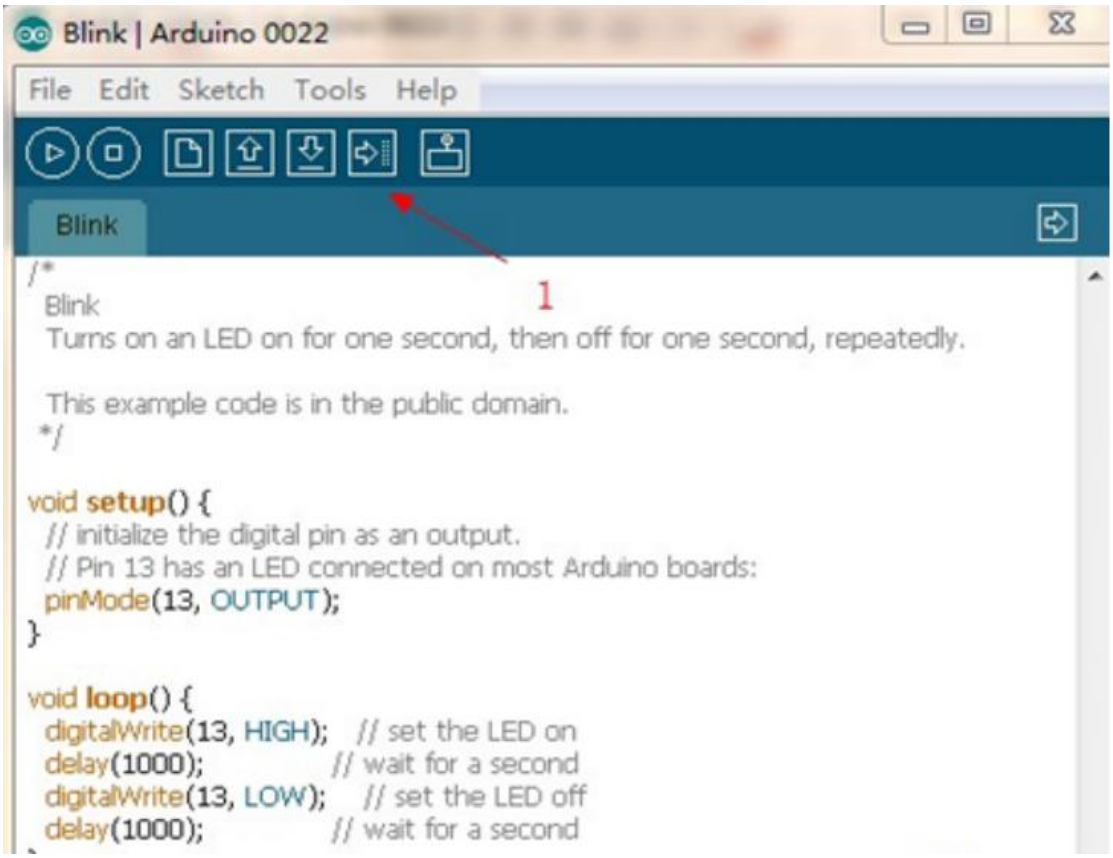


Next, import a sample program that makes the "L" light flash, and click [File] with the left mouse button.

Move the mouse to [Examples] in the pop-up drop-down menu. The menu expands to the right to [1.Basics]. After moving the mouse to [1.Basics], the menu continues to expand. Find [Blink] and left-click [Blink].



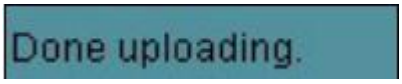
After clicking [Blink], an Arduino programming interface will pop up.



Directly click on the icon pointed to by the red arrow 1 in the left picture, and you will find that there will be two yellow lights on the Arduino motherboard that will flash for a while.

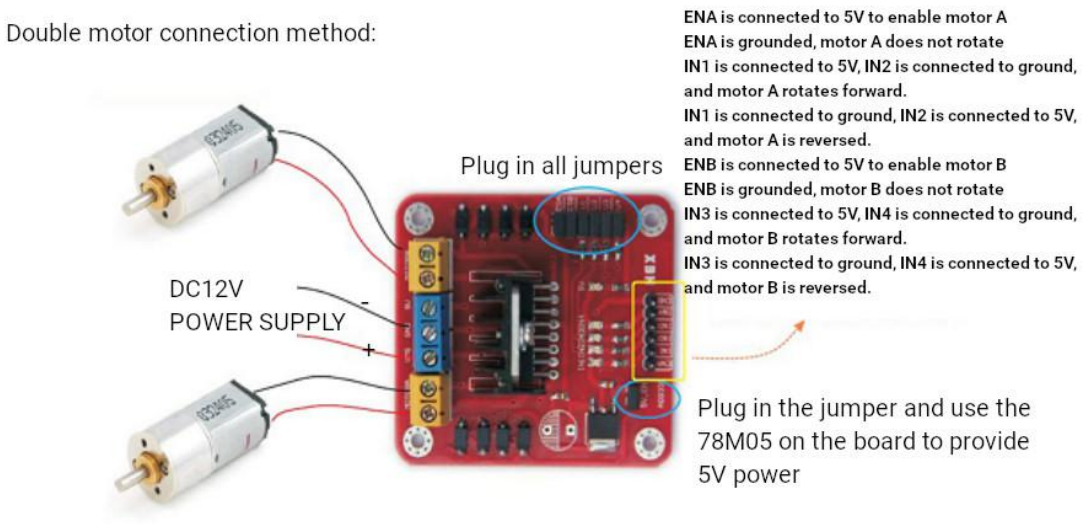
As the two flashing yellow lights went out. A text prompt appears below the programming box, and the L light on the motherboard turns on and off every second.

So congratulations, your first program has been successful! ! !



Experiment details

1. Application of L298N motor driver board



First of all, the VMS driver part can be powered by an external power supply, which is generally around 9V. The logic part can be powered by the board, that is, the terminals can be left floating or connected to +5V-+7V. The three pins in the left and right rows of the terminal are used to control two DC motors.

EA and EB are connected to the ArduinoPWM interface for motor speed regulation. The I1, I2, I3, and I4 interfaces are used to control the forward, backward, steering, and braking of the two DC motors respectively. They only need to be connected to the Arduino digital interface.

At this point, the preparation work is basically completed, and the program can be written. Here I have written the functions of the car to go straight, go backward, turn left, turn right, and brake into the program for your reference.

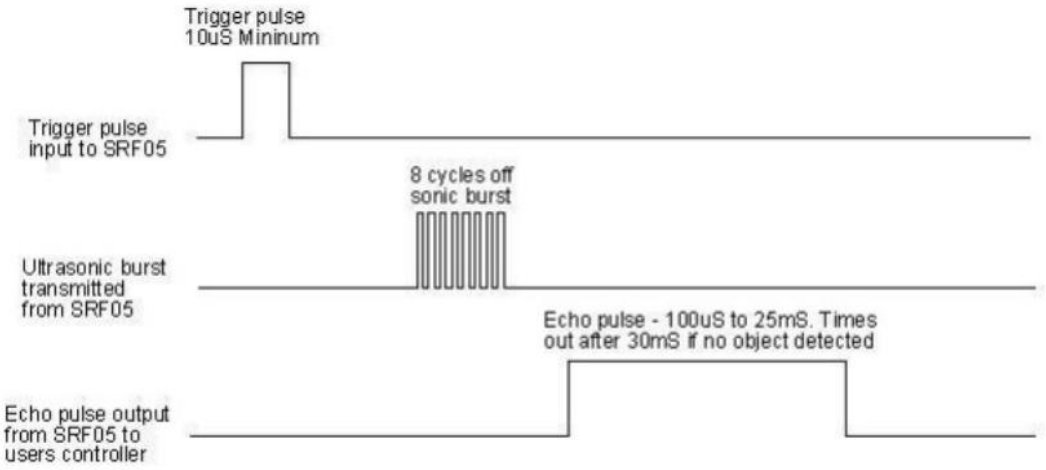
The procedure is as follows:

```
int pinI1=8;//define I1 interface
int pinI2=9;//Define I2 interface
int speedpin=11; //Define EA (PWM speed regulation) interface
int pinI3=6; //Define I3 interface
int pinI4=7;//Define I4 interface
int speedpin1=10; //Define EB (PWM speed regulation) interface
void setup()
{
  pinMode(pinI1,OUTPUT);
  pinMode(pinI2,OUTPUT);
  pinMode(speedpin,OUTPUT);
  pinMode(pinI3,OUTPUT);
  pinMode(pinI4,OUTPUT);
  pinMode(speedpin1,OUTPUT);
}
```

```
void loop()
{
//Go straight
analogWrite(speedpin,100);//Input analog value to set speed
   analogWrite(speedpin1,100);
   digitalWrite(pinI4,LOW);//Make the DC motor (right) rotate counterclockwise
   digitalWrite(pinI3,HIGH);
   digitalWrite(pinI1,LOW);//Make the DC motor (left) rotate clockwise
   digitalWrite(pinI2,HIGH);
delay(2000);
//Back
analogWrite(speedpin,100);//Input analog value to set speed
   analogWrite(speedpin1,100);
   digitalWrite(pinI4,HIGH);//Make the DC motor (right) rotate clockwise
   digitalWrite(pinI3,LOW);
digitalWrite(pinI1,HIGH);//Make the DC motor (left) rotate counterclockwise
   digitalWrite(pinI2,LOW);
delay(2000);
//Turn left
analogWrite(speedpin,60);//Input analog value to set speed
   analogWrite(speedpin1,60);
   digitalWrite(pinI4,LOW);//Make the DC motor (right) rotate counterclockwise
   digitalWrite(pinI3,HIGH);
   digitalWrite(pinI1,HIGH);//Make the DC motor (left) rotate counterclockwise
   digitalWrite(pinI2,LOW);
delay(2000);
//Turn right
analogWrite(speedpin,60);//Input analog value to set speed
   analogWrite(speedpin1,60);
   digitalWrite(pinI4,HIGH);//Make the DC motor (right) rotate clockwise
   digitalWrite(pinI3,LOW);
   digitalWrite(pinI1,LOW);//Make the DC motor (left) rotate clockwise
   digitalWrite(pinI2,HIGH);
delay(2000);
//brake
   digitalWrite(pinI4,HIGH);//Brake the DC motor (right)
digitalWrite(pinI3,HIGH);
   digitalWrite(pinI1,HIGH);//Brake the DC motor (left)
digitalWrite(pinI2,HIGH);
delay(2000);
}
```

Note: The left turn and right turn used in the program are only one control method for turning. The other methods are not listed one by one. You can try it yourself.

2. Ultrasonic ranging module



1. We first pull TRIG low, and then give at least a 10us high-level signal to trigger;
2. After triggering, the module will automatically transmit 8 40KHZ square waves and automatically detect whether there is a signal return.
3. If a signal returns, a high level is output through ECHO. The duration of the high level is the time from transmission to reception of the ultrasonic wave. Then the test distance = high level duration * 340m/s * 0.5;

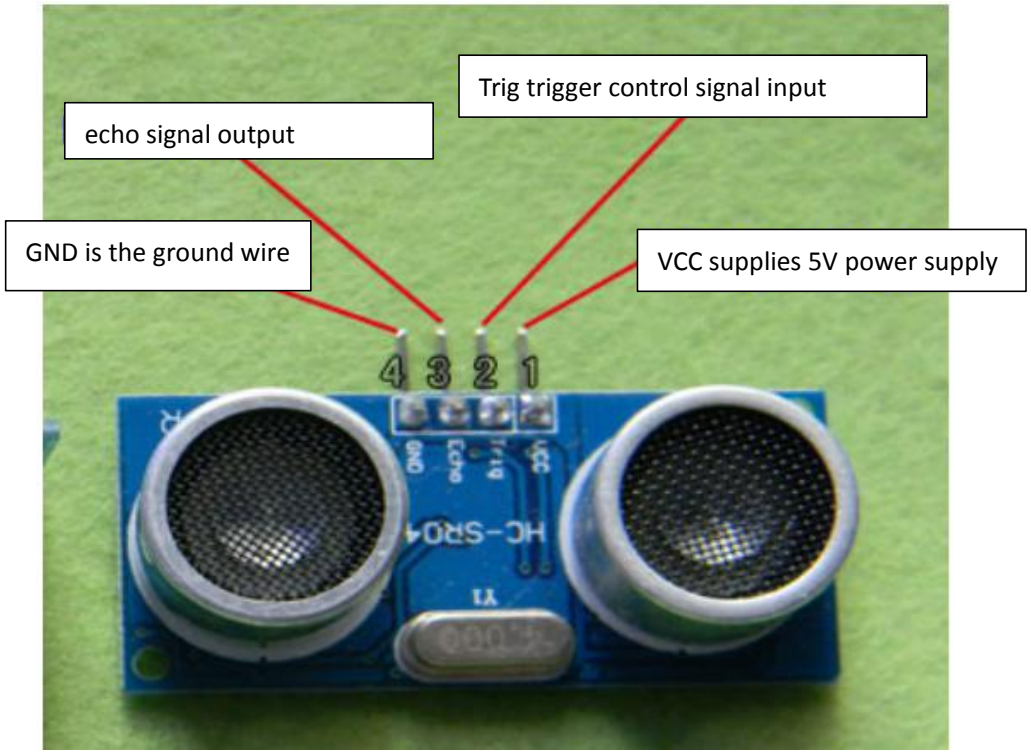
Electrical parameters

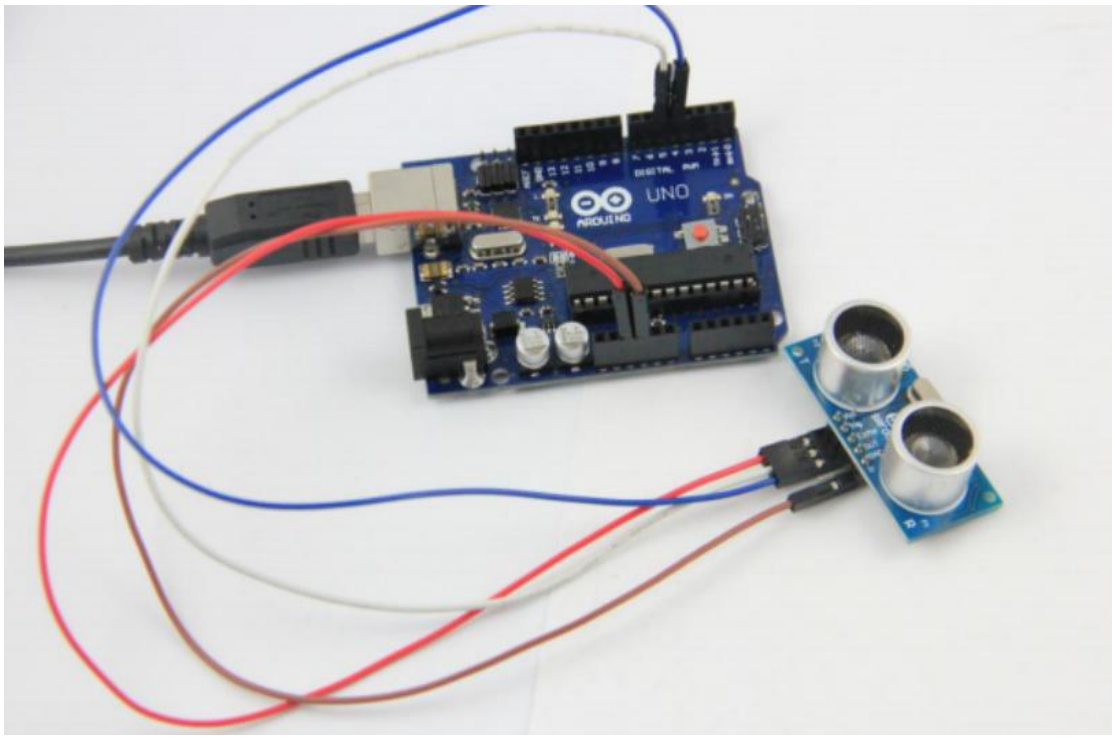
Working voltage: 0.5V (DC) Working current: 15mA

Detection distance: 2-450cm Detection angle: 15 degrees

Input trigger pulse: 10us TTL level

Output echo signal: Output TTL level signal (high), proportional to the range





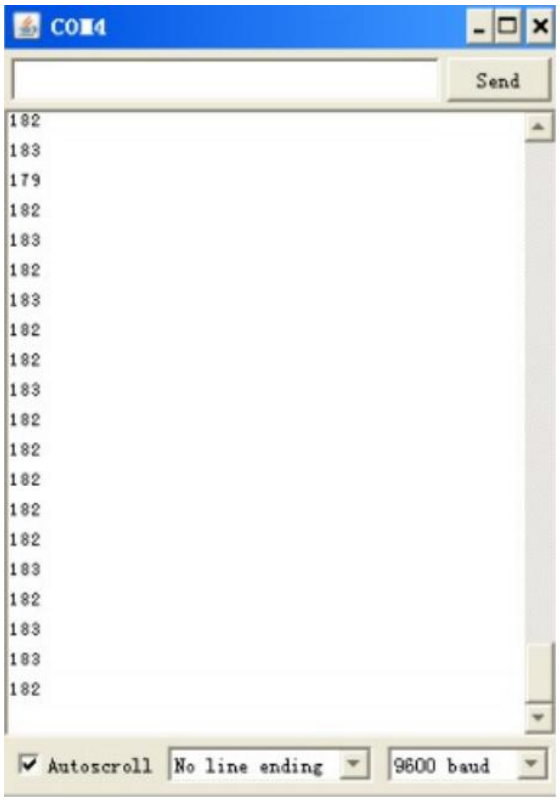
The D4 and D5 above refer to pins 4 and 5 of the digital port. There are specific physical connections below for reference. What we have to do this time is to learn how to use it to measure distance and use it on the computer. It is displayed on the display screen. Of course, if you want to make it look better, you can add an LCD or a digital tube. What we are doing here is a functional test, which is an inspiration process.

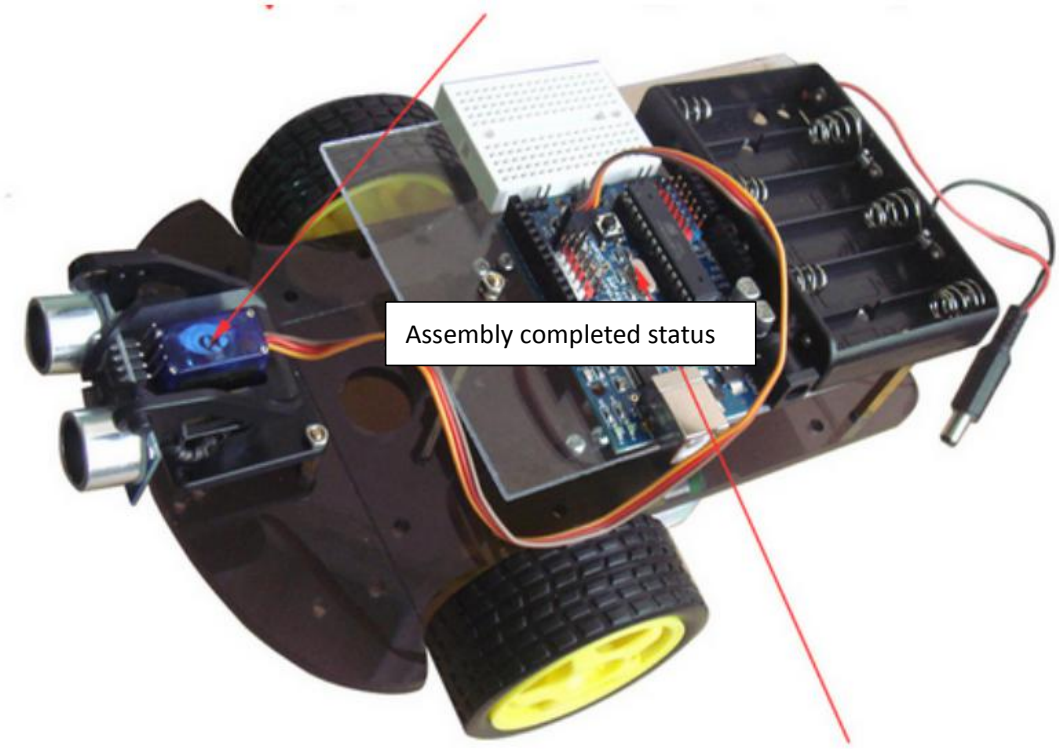
Well, below is our test code.

```
int inputPin=4; // define ultrasonic signal receiver pin ECHO to D4
int outputPin=5; // define ultrasonic signal transmitter pin TRIG to D5
void setup()
{
Serial.begin(9600);
pinMode(inputPin, INPUT);
pinMode(outputPin, OUTPUT);
}
void loop()
{
digitalWrite(outputPin, LOW);
delayMicroseconds(2);
digitalWrite(outputPin, HIGH); // Pulse for 10μ s to trigger ultrasonic
detection
delayMicroseconds(10);
digitalWrite(outputPin, LOW);
int distance = pulseIn(inputPin, HIGH); // Read receiver pulse time
distance= distance/58; // Transform pulse time to distance
Serial.println(distance); //Ourput distance
http://keyes-arduino.taobao.com
```

```
delay(50);
}
```

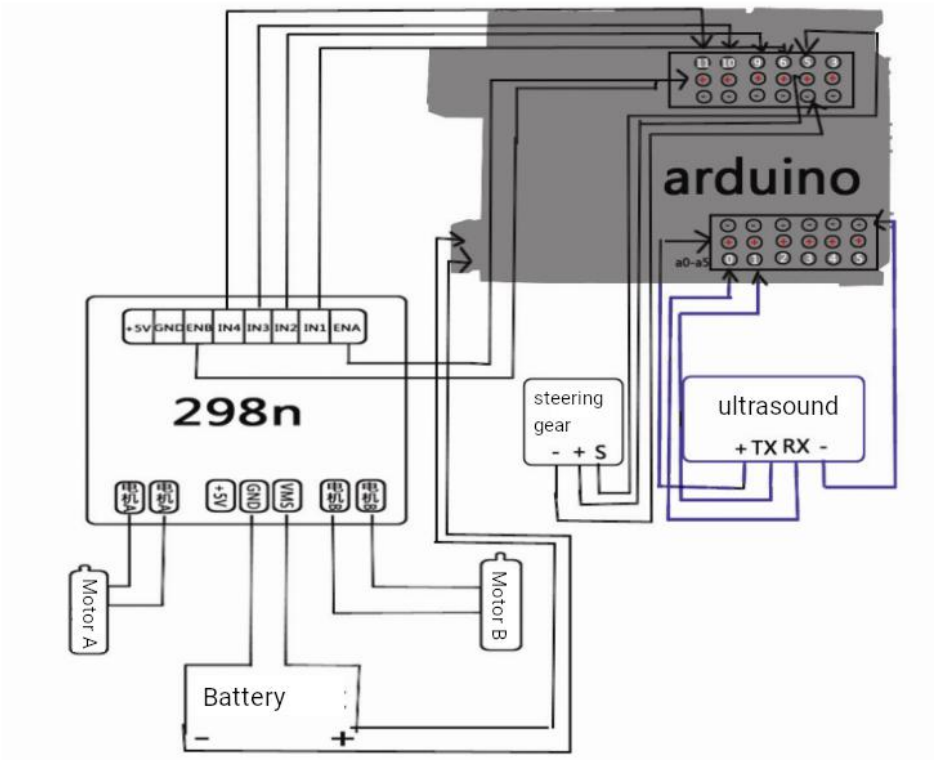
Of course, after compiling the above test code, downloading it to our Arduino control board, and opening the Serial Monito window, you will see a series of data displays, which is what we want, just like the one below.





Ultrasonic intelligent obstacle avoidance is convenient to implement, simple to calculate, easy to achieve real-time control, and can meet practical requirements in terms of measurement accuracy, so it has become a commonly used obstacle avoidance method. Reference for how to use ultrasonic waves (Arduino Ultrasonic Distance Measurement Instructions).

Ultrasonic smart wiring diagram;



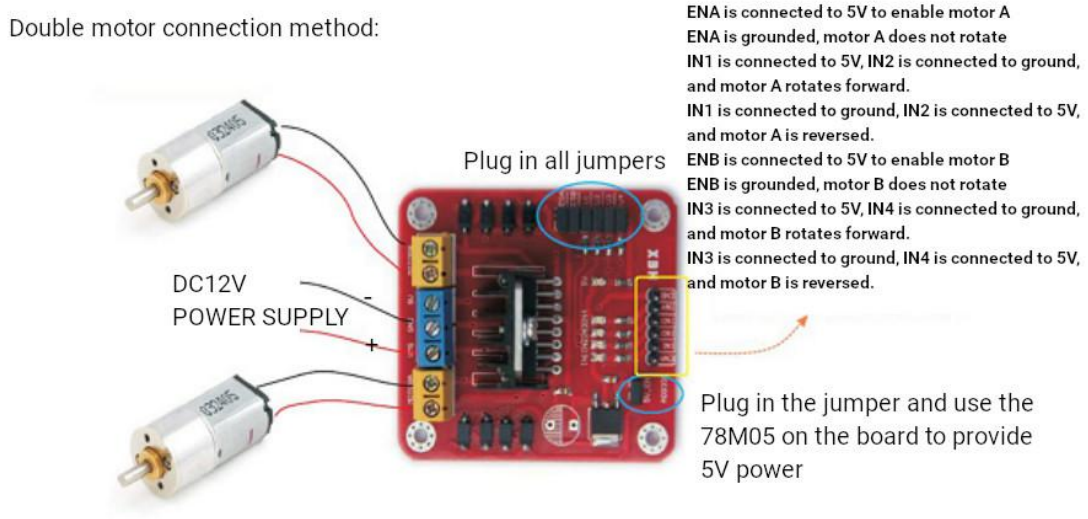
1: Motor connection

The motor is connected to the MOTOA of L298N

The second motor is connected to the MOTOB of L298N

2: Power supply processing of L298N

Use 6 No. 5 battery boxes to take one power supply to power the L298N motor drive module, and the other power supply to the ARDUINO motherboard. The + pole of the power supply that powers the L298N motor drive module is connected to the VMS interface of the L298N, and the - pole of the power supply is connected to the GND interface of the L298N. The +5V interface on the board is left open and not connected.



3: Motor enablement and steering (coordinating procedures)

int pinLB=6; // Define pin 6, left rear, connected to PWM6 pin of the force plate

int pinLF=9; // Define pin 9, left front, connected to the PWM9 pin of the force plate

int pinRB=10; // Define the right rear of pin 10 and connect it to the PWM10 pin of the force plate

int pinRF=11; // Define the right front of pin 11, connected to the PWM11 pin of the force plate

4: Connection of steering gear

myservo.attach(5); // Define servo motor output pin 5 (PWM)

5: Connection of ultrasonic sensor

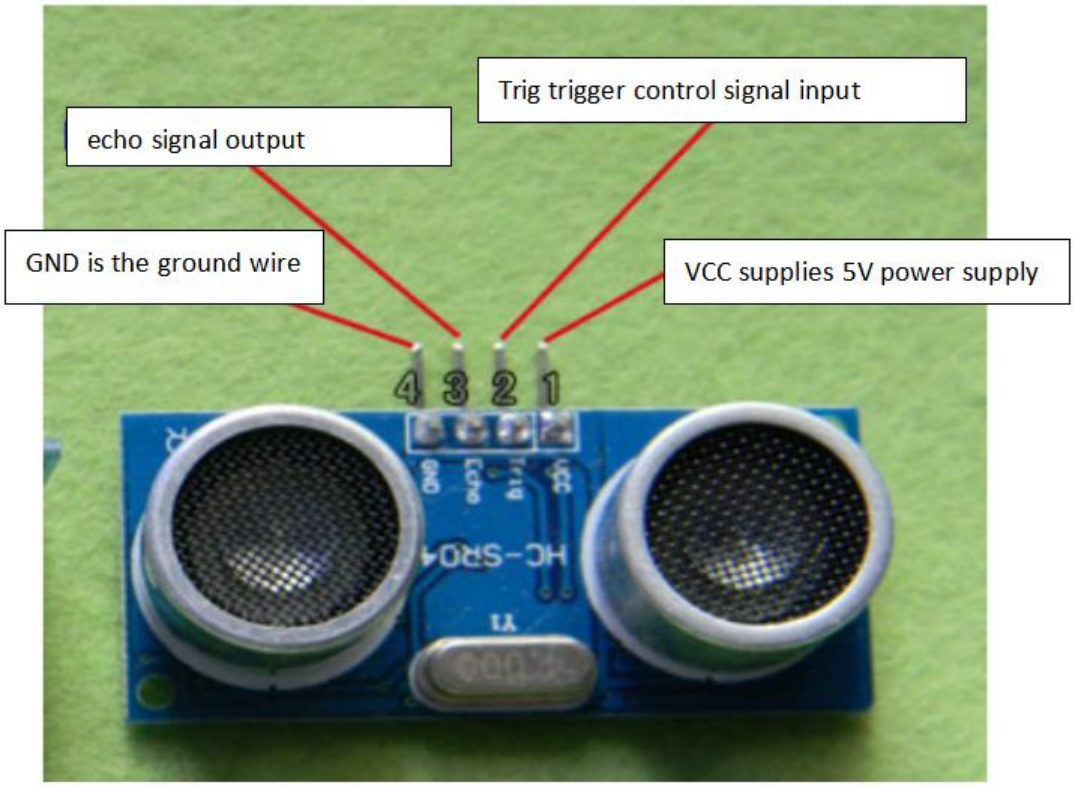
The ultrasonic sensor has four legs

VCC connected to +5V

TRIQ signal input

ECHO signal output

GND Ground




```
int inputPin = A0; // Define the ultrasonic signal receiving pin
int outputPin =A1; // Define the ultrasonic signal transmitting pin
Ultrasonic intelligent obstacle avoidance vehicle program (ARDUINO)
   L = left
   R = right
   F = front
   B = back
*/
#include <Servo.h>
int pinLB=6; //define pin 6 left rear
int pinLF=9; //define pin 9 left front
int pinRB=10; //define pin 10, right rear
int pinRF=11; //define pin 11, front right
int inputPin = A0; // Define the ultrasonic signal receiving pin
int outputPin =A1; // Define the ultrasonic signal transmitting pin
int Fspeedd = 0; // Front speed
int Rspeedd = 0; // Right speed
int Lspeedd = 0; // Left speed
int directionn = 0; // front=8 rear=2 left=4 right=6
Servo myservo; // Let myservo
int delay_time = 250; // Stabilization time after servo motor turns
int Fgo = 8; // Forward
int Rgo = 6; // turn right
int Lgo = 4; // turn left
```

```
int Bgo = 2; // Reverse
void setup()
{
   Serial.begin(9600); // Define the motor output pin
   pinMode(pinLB,OUTPUT); // Pin 8 (PWM)
   pinMode(pinLF,OUTPUT); // Pin 9 (PWM)
   pinMode(pinRB,OUTPUT); // Pin 10 (PWM)
   pinMode(pinRF,OUTPUT); // Pin 11 (PWM)

   pinMode(inputPin, INPUT); //Define the ultrasonic input pin
   pinMode(outputPin, OUTPUT); // Define the ultrasonic output pin
   myservo.attach(5); // Define servo motor output pin 5 (PWM)
}
void advance(int a) // Forward
{
   digitalWrite(pinRB,LOW); // Activate the motor (rear right)
   digitalWrite(pinRF,HIGH);
   digitalWrite(pinLB,LOW); // Activate the motor (rear left)
   digitalWrite(pinLF,HIGH);
   delay(a * 100);
   }
void right(int b) //turn right (single wheel)
   {
   digitalWrite(pinRB,LOW); // Make the motor (right rear) move
   digitalWrite(pinRF,HIGH);
   digitalWrite(pinLB,HIGH);
   digitalWrite(pinLF,HIGH);
   delay(b * 100);
   }
void left(int c) //Turn left (single wheel)
   {
   digitalWrite(pinRB,HIGH);
   digitalWrite(pinRF,HIGH);
   digitalWrite(pinLB,LOW); // Make the motor (rear left) move
   digitalWrite(pinLF,HIGH);
   delay(c * 100);
   }
void turnR(int d) //Turn right (two wheels)
   {
   digitalWrite(pinRB,LOW); // Make the motor (right rear) move
   digitalWrite(pinRF,HIGH);
   digitalWrite(pinLB,HIGH);
   digitalWrite(pinLF,LOW); // Make the motor (front left) move
   delay(d * 100);
```

```
   }
void turnL(int e) //Turn left (two wheels)
   {
   digitalWrite(pinRB,HIGH);
   digitalWrite(pinRF,LOW); // Make the motor (front right) move
   digitalWrite(pinLB,LOW); // Make the motor (rear left) move
   digitalWrite(pinLF,HIGH);
   delay(e * 100);
   }
void stopp(int f) //Stop
   {
   digitalWrite(pinRB,HIGH);
   digitalWrite(pinRF,HIGH);
digitalWrite(pinLB,HIGH);
   digitalWrite(pinLF,HIGH);
   delay(f * 100);
   }
void back(int g) //Back
   {
   digitalWrite(pinRB,HIGH); // Make the motor (right rear) move
   digitalWrite(pinRF,LOW);
   digitalWrite(pinLB,HIGH); // Make the motor (left rear) move
   digitalWrite(pinLF,LOW);
   delay(g * 100);
   }

void detection() //Measure 3 angles (0.90.179)
   {
   int delay_time = 250; // Stabilization time after servo motor turns
   ask_pin_F(); // Read the distance ahead

   if(Fspeedd < 10) // If the distance ahead is less than 10 cm
   {
   stopp(1); // Clear output data
   back(2); // Go back 0.2 seconds
   }

   if(Fspeedd < 25) // If the distance ahead is less than 25 cm
   {
   stopp(1); // Clear output data
   ask_pin_L(); // Read the distance to the left
   delay(delay_time); // Wait for the servo motor to stabilize
   ask_pin_R(); // Read the distance to the right
   delay(delay_time); // Wait for the servo motor to stabilize
```

```
   if(Lspeedd > Rspeedd) //If the left distance is greater than the right distance
   {
   directionn = Rgo; //Go right
   }

   if(Lspeedd <= Rspeedd) //If the left distance is less than or equal to the right distance
   {
   directionn = Lgo; //Go left
   }

   if (Lspeedd < 10 && Rspeedd < 10) //If the left distance and right distance are both less than
10 cm
{
   directionn = Bgo; //Go backward
   }
   }
   else //Add if the front is not less than (greater than) 25 cm
   {
   directionn = Fgo; //go forward
   }

   }
void ask_pin_F() // Measure the distance ahead
   {
   myservo.write(90);
   digitalWrite(outputPin, LOW); // Let the ultrasonic wave emit low voltage for 2 μ s
   delayMicroseconds(2);
   digitalWrite(outputPin, HIGH); // Let the ultrasonic wave emit high voltage for 10 μ s, here it is
at least 10 μ s
   delayMicroseconds(10);
   digitalWrite(outputPin, LOW); // Maintain ultrasonic emission low voltage
   float Fdistance = pulseIn(inputPin, HIGH); // Read the difference time
   Fdistance= Fdistance/5.8/10; // Convert time to distance (unit: centimeters)
   Serial.print("F distance:"); //Output distance (unit: centimeters)
   Serial.println(Fdistance); //Display distance
   Fspeedd = Fdistance; // Read the distance into Fspeedd (front speed)
   }
void ask_pin_L() // Measure the distance to the left
   {
   myservo.write(5);
   delay(delay_time);
   digitalWrite(outputPin, LOW); // Let the ultrasonic wave emit low voltage for 2 μ s
   delayMicroseconds(2);
```

```
  digitalWrite(outputPin, HIGH); // Let the ultrasonic wave emit high voltage for 10 μ s, here it is at least 10 μ s
  delayMicroseconds(10);
  digitalWrite(outputPin, LOW); // Maintain ultrasonic emission low voltage
  float Ldistance = pulseIn(inputPin, HIGH); // Read the difference time
  Ldistance= Ldistance/5.8/10; // Convert time to distance (unit: centimeters)
  Serial.print("L distance:"); //Output distance (unit: centimeters)
  Serial.println(Ldistance); //Display distance
  Lspeedd = Ldistance; // Read the distance into Lspeedd (left speed)
  }
void ask_pin_R() // Measure the distance to the right
  {
  myservo.write(177);
  delay(delay_time);
  digitalWrite(outputPin, LOW); // Let the ultrasonic wave emit low voltage for 2 μ s
delayMicroseconds(2);
  digitalWrite(outputPin, HIGH); // Let the ultrasonic wave emit high voltage for 10 μ s, here it is at least 10 μ s
  delayMicroseconds(10);
  digitalWrite(outputPin, LOW); // Maintain ultrasonic emission low voltage
  float Rdistance = pulseIn(inputPin, HIGH); // Read the difference time
  Rdistance= Rdistance/5.8/10; //Convert time to distance (unit: centimeters)
  Serial.print("R distance:"); //Output distance (unit: centimeters)
  Serial.println(Rdistance); //Display distance
  Rspeedd = Rdistance; // Read the distance into Rspeedd (right speed)
  }

void loop()
{
  myservo.write(90); //Return the servo motor to the ready position to prepare for the next measurement
  detection(); //Measure the angle and determine which direction to move

  if(directionn == 2) //if directionn(direction) = 2(reverse)
  {
  back(8); // Back(car)
  turnL(2); //Move slightly to the left (to prevent getting stuck in a blind alley)
  Serial.print(" Reverse "); //Display direction (reverse)
  }
  if(directionn == 6) //if directionn(direction) = 6(turn right)
  {
  back(1);
  turnR(6); // turn right
  Serial.print(" Right "); //Display direction (turn left)
```

```
    }
    if(directionn == 4) //if directionn(direction) = 4(turn left)
    {
    back(1);
    turnL(6); // turn left
    Serial.print(" Left "); //Display direction (turn right)
    }
    if(directionn == 8) //if directionn(direction) = 8(forward)
    {
    advance(1); //Normal advance
    Serial.print(" Advance "); //Display direction (forward)
    Serial.print(" ");
    }
}
```