

Lesson

6

Bluetooth Control



Introduction:

There are three parts in this lesson. We need to learn how to send instructions to our car via Bluetooth so that we can control the car according to our ideas.

Bluetooth Control

- 1.1 Hardware Design
- 1.2 Software Design
- 1.3 Upload Validation

Bluetooth Control LED

- 2.1 Software Design
- 2.2 Upload Validation

Preparations:

- one car (with a battery)
- one USB cable

1.1 Hardware Design

Bluetooth: Short distance wireless communication transmission function. It includes hardware and software communication protocol. That is, the Bluetooth of the two terminals can receive and send data within a certain range.



In the Kit, we use "DX-BT16" Bluetooth module model. (As shown in figure 1.1.1)

DX-BT16 communicates with Nano through the RX/TX pin. (As shown in figure 1.1.2)



Figure 1.1.1 The Real Object of DX-BT16

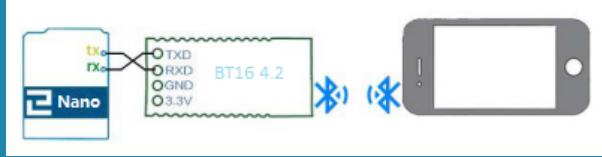


Figure 1.1.2 Communication flow chart

The description of Bluetooth module:

1. Adopt mainstream Bluetooth chip of TI, protocol standard of BluetoothV4.2.
2. Analog working voltage of serial port is 3.3V.
3. Users can set baud rate as 1200, 2400, 4800, 9600, 19200, 38400, 57600 and 115200.
4. Dimension of key module is 18mm x 14.5 mm x2 mm.
5. Working electric current: 40MA.
6. Dormancy electric current: less than1MA.
7. Being used for GPS navigation system, hydroelectric gas reading system and industrial field mining control system.
8. Can be connected to laptop with Bluetooth support, computer with Bluetooth adapter, PDA, etc.

1.2 Software Design

Please open ELEGOO Tumbler Self-Balancing Car Tutorial -> Lesson 6 Bluetooth Control -> Bluetooth_Control-> Bluetooth_Control.ino in the current folder.

Serial.available(): Judge whether there is data in serial port.

Serial.read(): Read one byte data at a time.

```
void Bluetooth_Data_Processing()
{
    if (Serial.available())
    {
        char data = Serial.read();
        Serial.print("data:");Serial.println(data);
        if (data != '\0' && data != '\n')
        {
            switch (data)
            {
                case '1': Serial.println("1 is pressed by you"); break;

                case '2': Serial.println("2 is pressed by you"); break;

                case '3': Serial.println("3 is pressed by you"); break;

                case '4': Serial.println("4 is pressed by you"); break;

                case '5': Serial.println("5 is pressed by you"); break;

                case '6': Serial.println("6 is pressed by you"); break;
            }
        }
    }
}
```

1.3 Upload Validation

First, upload the program.

Open the APP

As below, we use an iPhone for example to show you how to control the Elegoo Tumbller via this App:

STEP1: First of all, turn on your cellphone's Bluetooth function.

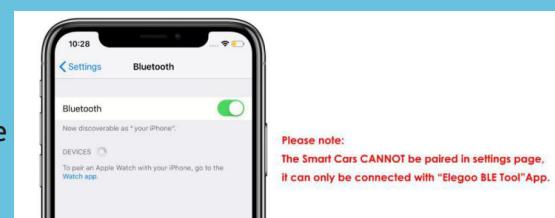
STEP2:

Open the "Elegoo BLE Tool" App.

Click OK. (Please click OK when App requests for the location permission, otherwise it will affect the Bluetooth function)

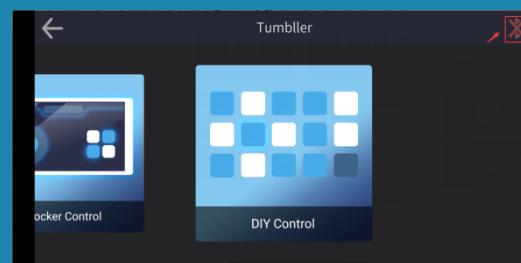
STEP3: Choose the language

First, run the installed App, and click the icon in the top left corner to enter the language list.



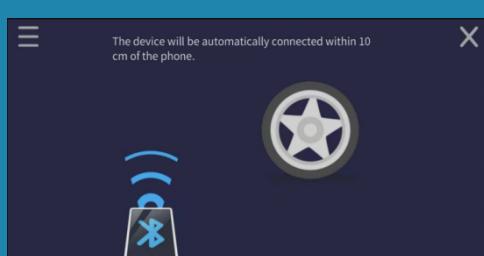
STEP4: Connect Bluetooth (Turn on the power in the car, but don't disconnect the USB cable)

Click Tumbller to enter the control page. Then tap the " " icon to enter the Bluetooth searching interface.

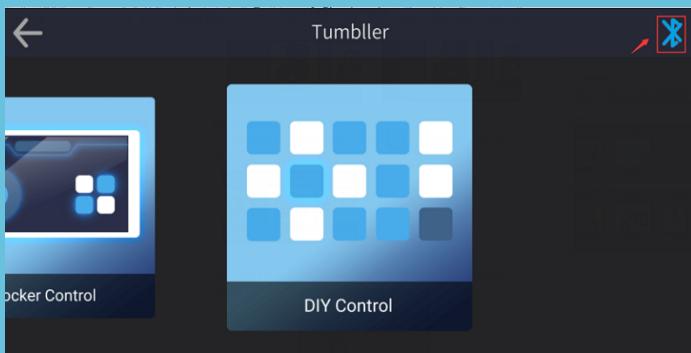


Put your phone close to the Tumbller (within 10cm), the app will connect to the Tumbller automatically.

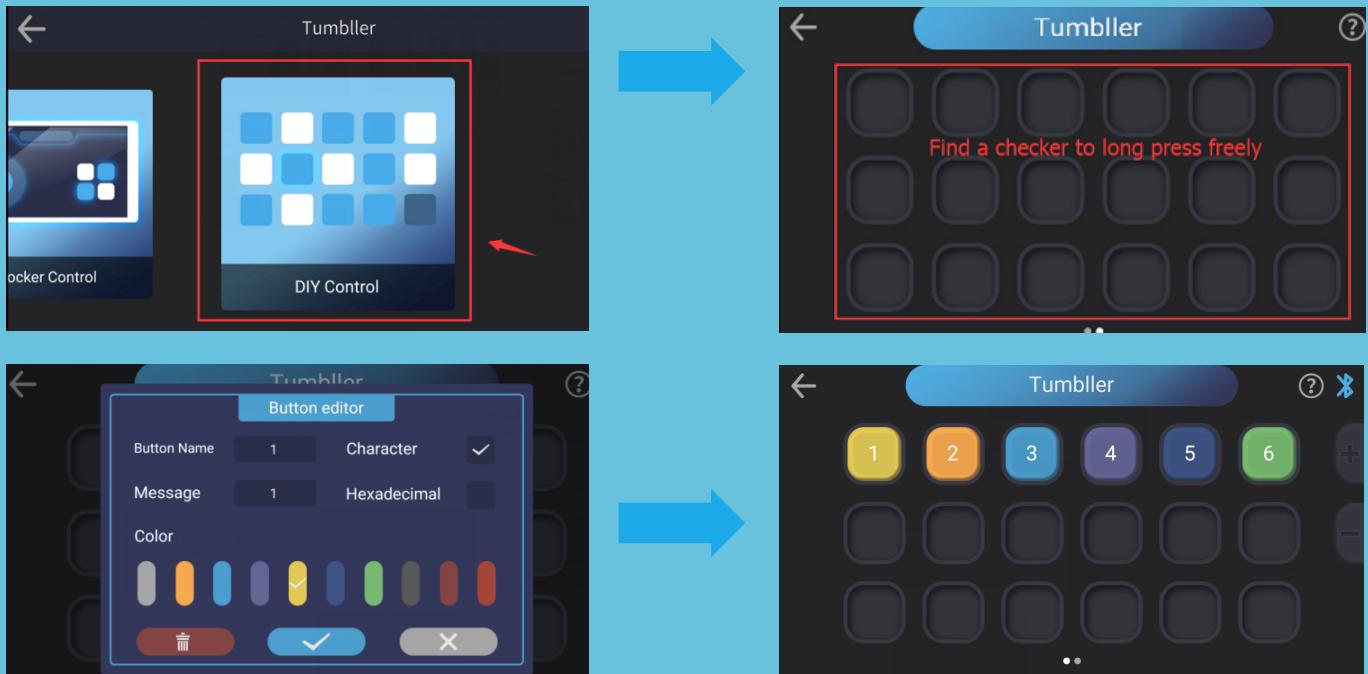
You can also open the Bluetooth device list by tapping the menu icon in the upper left corner and select "ELEGOO BT16" to connect the Tumbller manually.



The Bluetooth status icon will turn blue when the Tumbller is connected.



Open the mobile APP, and set up parameters as follows.



Open the serial port, and then press these buttons, and you will find that the corresponding information will be printed out.

```
data:1
1 is pressed by you
data:2
2 is pressed by you
data:3
3 is pressed by you
data:4
4 is pressed by you
data:5
5 is pressed by you
data:6
6 is pressed by you
```

Bluetooth Control LED

2.1 Software Design

Since we can print out the information we want through the Bluetooth connection setting button, we can also do the things we want through the Bluetooth setting button, such as switching the light mode.

Please open [ELEGOO Tumbler Self-Balancing Car Tutorial](#) -> [Lesson 6 Bluetooth Control](#) -> [Bluetooth_Control_LED->Bluetooth_Control_LED.ino](#) in the current folder.

Change the corresponding printing information to light effect directly.

```
// in Rgb.h
void Bluetooth_Data_Processing()
{
    if (Serial.available())
    {
        char data = Serial.read();
        switch (data)
        {
            case '1':theaterChaseRainbow(50);break;
            case '2':rainbowCycle(20);break;
            case '3':theaterChase(127, 127, 127, 50);break;
            case '4':rainbow(20);break;
            case '5':whiteOverRainbow(20, 30, 4);break;
            case '6':rainbowFade2White(3, 50, 50);break;
        }
    }
}
```

Bluetooth_Data_Processing()

However, it should be noted that delay () in the light effect will cause failure to switch the light effect in time, so we need to call Bluetooth_Data_Processing() again after delay().

```
// in Rgb.h
bool theaterChaseRainbow(uint8_t wait)
{.....
delay(wait);Bluetooth_Data_Processing();
.....
}
bool rainbowCycle(uint8_t wait) {.....
delay(wait);Bluetooth_Data_Processing();
.....
}
bool theaterChase(uint8_t r, uint8_t g, uint8_t b, uint8_t t wait)
{.....
delay(wait);Bluetooth_Data_Processing();
.....
}
bool rainbow(uint8_t wait)
{.....
delay(wait);Bluetooth_Data_Processing();
.....
}
bool whiteOverRainbow(uint8_t wait, uint8_t whiteSpeed, uint8_t whiteLength)
{.....
delay(wait);Bluetooth_Data_Processing();
.....
}
bool rainbowFade2White(uint8_t wait, int rainbowLoops, int whiteLoops)
{.....
delay(wait);Bluetooth_Data_Processing();
.....
}
```

2.2 Upload Validation

Upload the program. Turn on the power switch and press the key again to switch different lighting effects.

