

## Pirate Ship --- Bluetooth APP Control

As the robot has been enabled to perform the basic tasks, we now upgrade it with one more feature, namely, control the robot from your cellphone via Bluetooth. Also, you can add more cool lighting effects to your robot. Combined with our new APP, I believe you will love it so much. At the same time, we also have been optimized code for Bluetooth control ,that will more convenient for you to use it.

### Preparation

- iPhone or iPad × 1
- GoBLE APP × 1



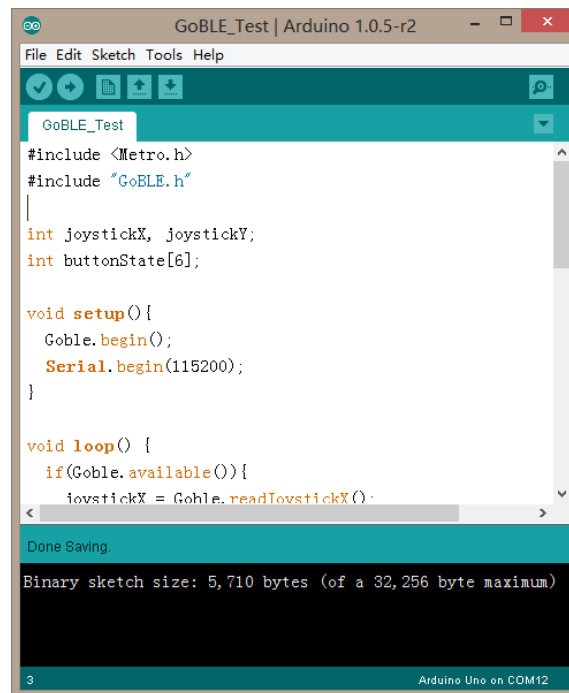
**Note:** You should at least have iOS 8.1 installed on your phone. Since optimized, you can easily enjoy the app on iPhone 5, iPhone 6 or iPhone 6 Plus.

**Currently, we have only developed the APP for iOS, but it is not available for Android.** For iPhone users: please download GoBLE from the App Store. If the screen is shown as the photo below, Congratulations! You've got the right APP!



## Upload the Testing Code

Please connect the your Romeo BLE board to your computer by USB cable. The testing code named GoBLE\_Test.ino can be found in the software package. It only enables you to check the signal from your iPhone on PC. DOWNLOAD IT & CHECK IT OUT. Don't forget to load GoBLE library before you download the testing code. Once downloaded, the "Done uploading" notice would pop up.

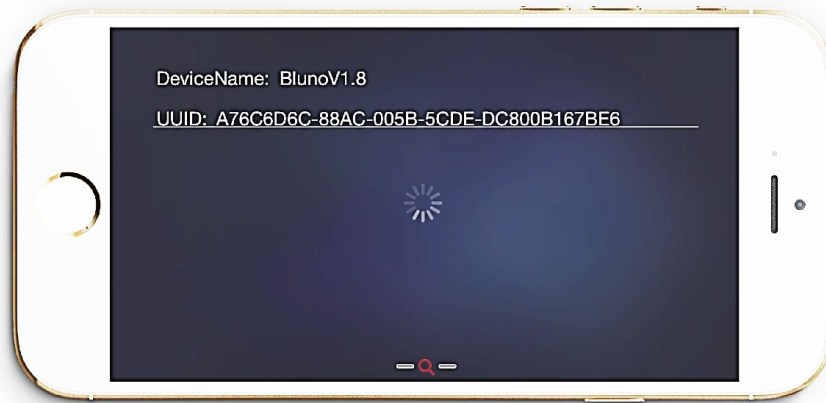


## Bluetooth Test

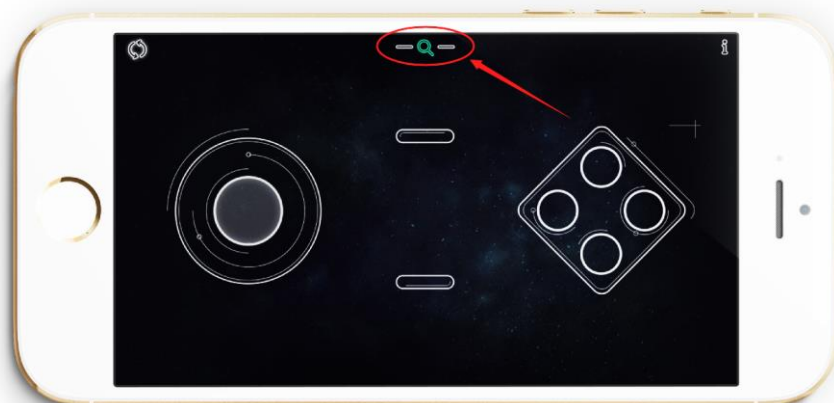
If you have finished upload the testing code, now let's start the Bluetooth pairing in order to connect your robot to iPhone. Firstly, remember to turn on the mobile bluetooth ,and then open the GoBLE app. Now, your screen maybe look like this!



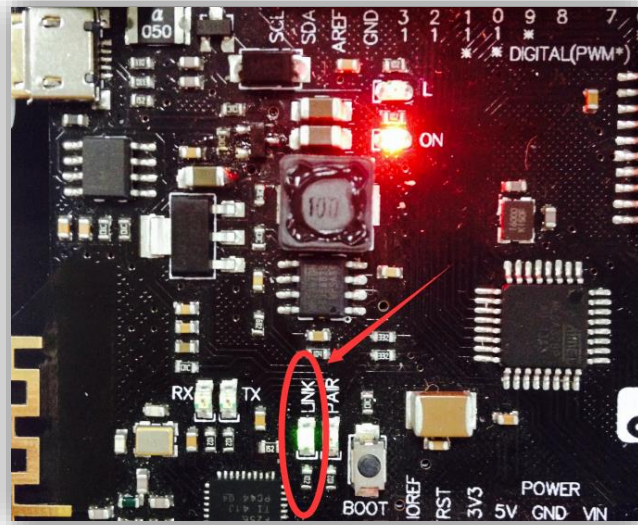
Click the Bluetooth search key(it looks like the red magnifying glass).Now, it will show you all the Bluetooth devices listed on the screen. Then, you only need to choose one of them that corresponds to the name of Romeo BLE board. Click the UUID and they will start pairing.



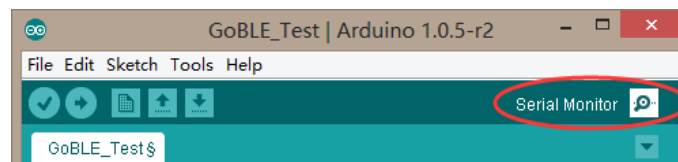
Once connected, the color of Bluetooth search key will change from red to green just as it is shown in the picture below.



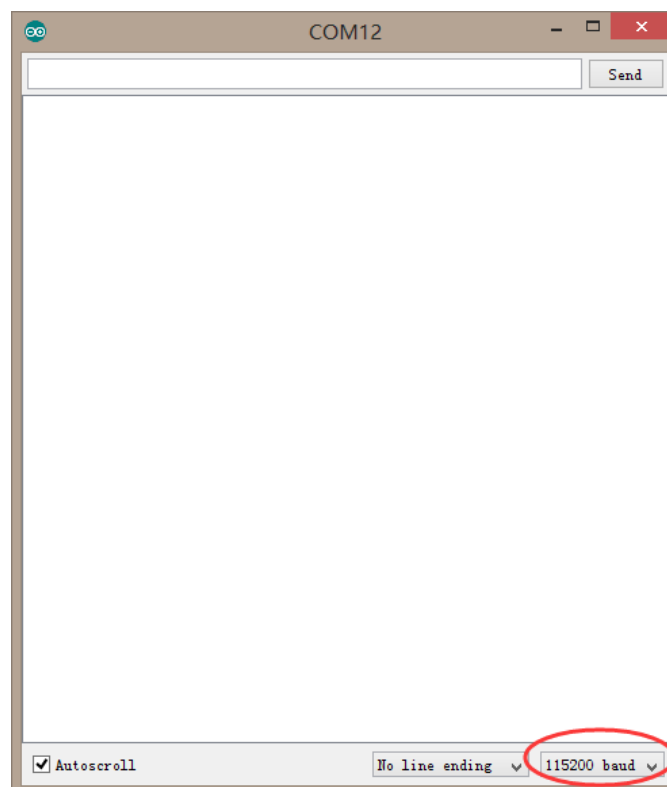
At the same time, the LINK led on the board will turn on, that means your iPhone have been connected to the Romeo BLE controller.



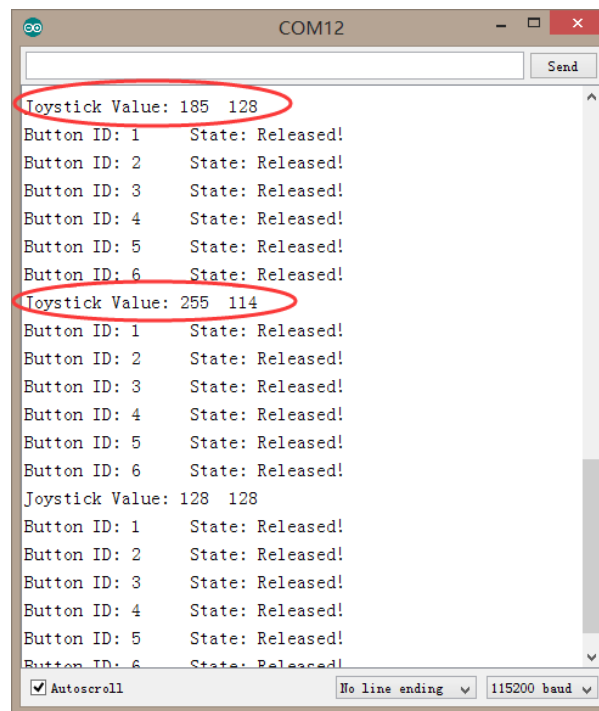
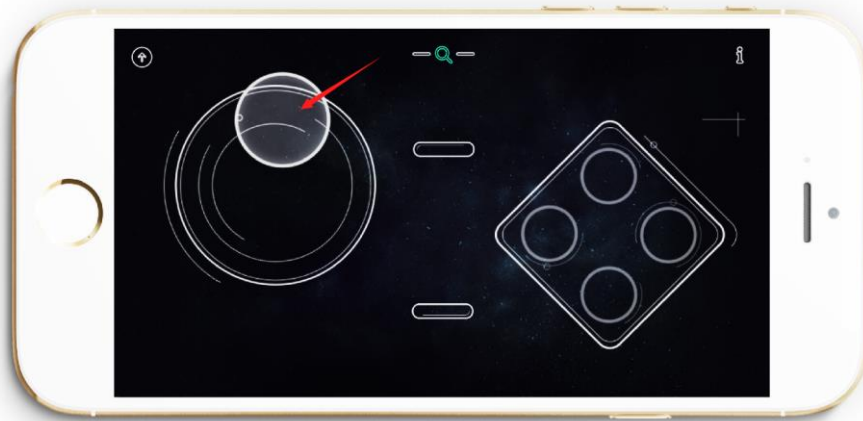
Now let's test if the controller can successfully receive signal from your iPhone. Go back to the Arduino IDE interface and click the Serial Monitor on the upper right-hand corner.

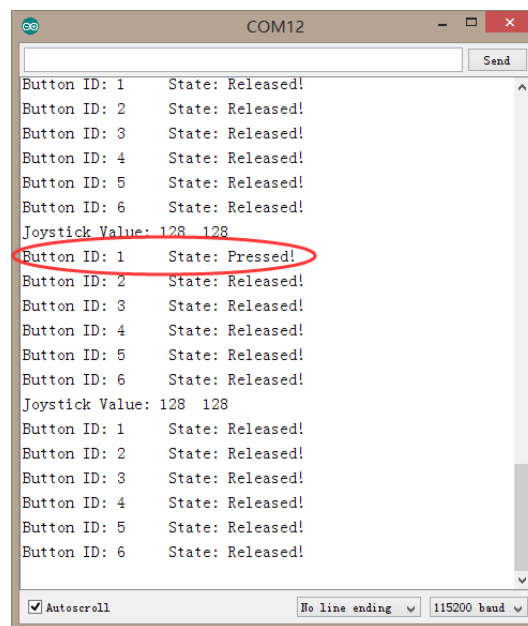
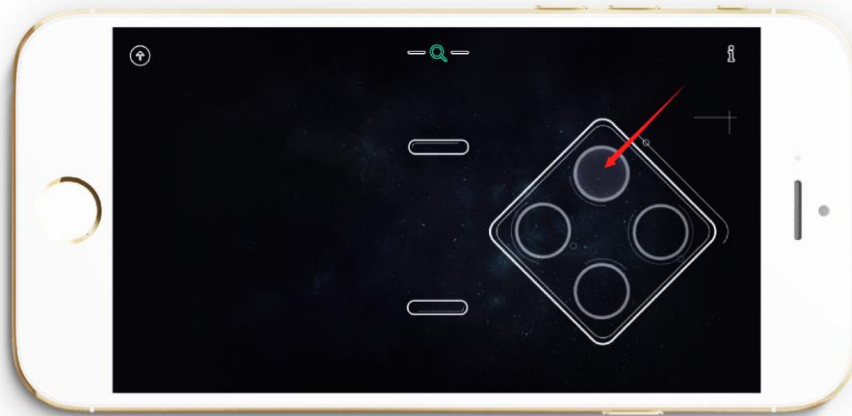


A dialogue as it is shown in the picture below would pop up. Firstly, the baud rate of serial port shall be set to be 115200 as it is shown in the red circle below.



Now you can take out your iPhone, and randomly swipe the virtual joystick or click the button on the screen. If nothing else, you will see the corresponding value on the Serial Monitor. And joystick value on the Serial Monitor will be changed when you swipe the virtual joystick on the screen. You will find that the value increase when joystick swipe up, or the value decrease when the joystick swipe down. Did you find that? Button ID means which button you are clicking.





## Upload the Code for Remote Control

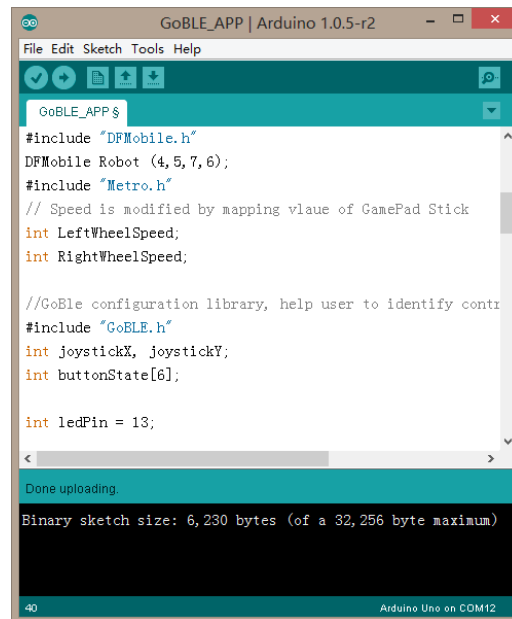
If everything is ok, let's continue and upload the code for remote controlling to Arduino board. There is a very key point to keep in your mind!

Your iPhone has already been connected to Romeo BLE Board in the last testing stage. However, you are unable to upload the code to Romeo BLE board when the Bluetooth has been established between the your phone and the board. Thus, you should be BREAK OFF the Bluetooth pairing first. In addition, you could upload the code again if the board has been disconnected from the phone. THIS POINT IS VERY IMPORTANT! You need to remember that.

There are two methods to interrupt the connection.

One way is that you can simply turn off the Remeo BLE power and then turn it on again. The other way is that just turn off the Bluetooth on your phone.

After disconnected, you can upload the example code again. The code is named as GoBLE\_APP.ino in the code package.



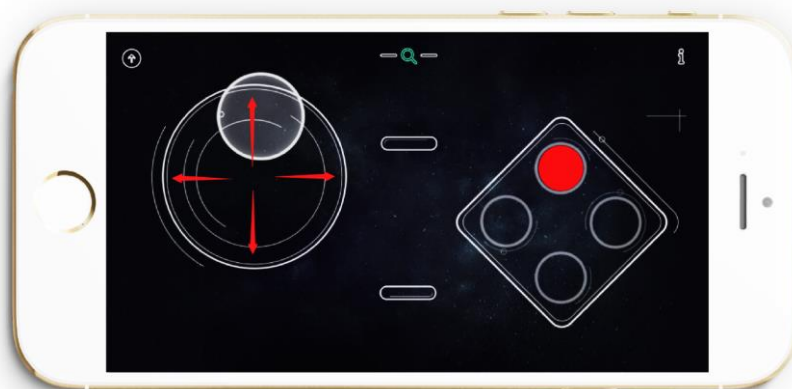
```
GoBLE_APP | Arduino 1.0.5-r2
File Edit Sketch Tools Help
GoBLE_APP $
#include "DFMobile.h"
DFMmobile Robot (4,5,7,6);
#include "Metro.h"
// Speed is modified by mapping vlaue of GamePad Stick
int LeftWheelSpeed;
int RightWheelSpeed;

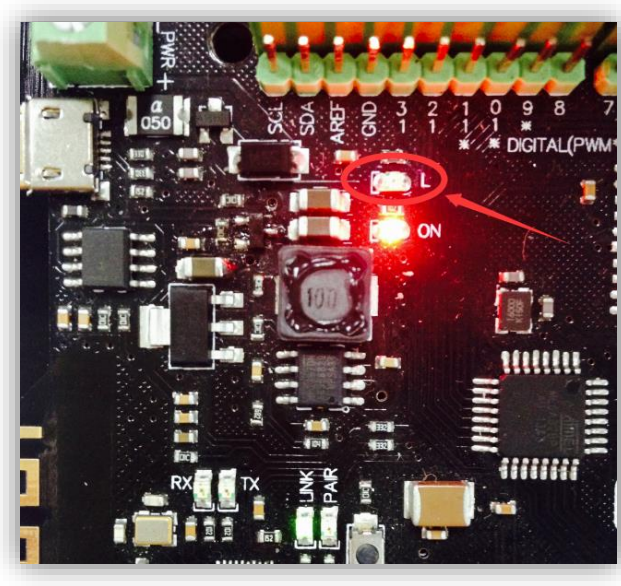
//GoBLE configuration library, help user to identify contr
#include "GoBLE.h"
int joystickX, joystickY;
int buttonState[6];

int ledPin = 13;

Done uploading.
Binary sketch size: 6,230 bytes (of a 32,256 byte maximum)
40 Arduino Uno on COM12
```

Once uploaded successfully, you may follow the red arrow in the picture below and swipe the Joystick to control your robot. Also, you may click the red button to turn on or turn off LED on the board which is connected with PIN13.





## Code Synopsis

After playing the robot, let's start learning how to use the code. No more words for the testing code for GoBLE test. Then we just move ahead to the GoBLE APP.

There's no need to discuss the basic code – let's just take a look at the part involving bluetooth control. If you want to remote control the robot, two libraries named Metro and GoBLE will be used.

```
#include "Metro.h"
#include "GoBLE.h"
int joystickX, joystickY;
int buttonState[6];
```

joystickX, joystickY and buttonState[6], three variables, are defined for the GoBLE library. They are used to store the state value for the X-axis, Y-axis and those values of six buttons.

The map below shows the layout of all buttons. You may probably know the meanings of  $+X$ ,  $-X$ ,  $+Y$ , and  $-Y$  if you have analyzed the change of the state values.

X & Y are the Joystick's directions of movement."+, -" show the trends of the value. "+" means the state value is increasing. And "-" means the state value is decreasing.





Initial setting shall be written in the function of Setup().

```
Goble.begin();
```

This line is used for initiating the bluetooth setup. And it cannot be missed whenever you use the Bluetooth code on your board.

```
Serial.begin(115200);
```

In fact, this line has been used at the testing stage. It is used to initiating the serial port. If you want to read the value from the Serial Monitor, you must write this statement in the setup function. AND , also need to set the baud rate.

Because the baud rate of Bluetooth is special, you need to set to 115200. Make sure the baud rate has been setup ,otherwise it may cause some problems.

Please continue down look . The next line is the function of Goble.available()

```
if(Goble.available()){  
    do something;  
}
```

Meaning that what the next action is once the figure has been received via Bluetooth.

What's written within the brackets would be our next action. First, we need to analyze the figure received. The following two statement is to read the value on X and Y axis.

```
joystickX = Goble.readJoystickX();
joystickY = Goble.readJoystickY();
// Serial.print("joystickX: ");
// Serial.print(joystickX);
// Serial.print("joystickY: ");
// Serial.println(joystickX);
```

Guess what the four statements `Serial.print()` above mean. It is related to the Serial. It's for the serial to print the data received, which is convenient for code debug and optimization.

`///  
//` means annotation of the following content. These four sentences won't be included when compiling the code. Namely, no data will be sent to the serial once we use the annotate for these four statements.

Please refer to the Reference page of Arduino Website([www.arduino.cc](http://www.arduino.cc)) for further information.

```
buttonState[SWITCH_UP]      = Goble.readSwitchUp();
buttonState[SWITCH_DOWN]    = Goble.readSwitchDown();
buttonState[SWITCH_LEFT]    = Goble.readSwitchLeft();
buttonState[SWITCH_RIGHT]   = Goble.readSwitchRight();
buttonState[SWITCH_SELECT]  = Goble.readSwitchSelect();
buttonState[SWITCH_START]   = Goble.readSwitchStart();
```

All contents above are used to load the information about the state of the buttons. Layout of the buttons is as follows.

```
SWITCH_UP      -- 1
SWITCH_RIGHT   -- 2
SWITCH_DOWN    -- 3
SWITCH_LEFT    -- 4
SWITCH_SELECT  -- 5
SWITCH_START   -- 6
```



We need to process all the data read before we starting using them.

The value read on the Joystick shall be mapped to the rotational speed of the wheels of our robot. Thus, rotational speed of the wheels turns out to be between -255 and 255.

```
int SpeedX=2*joystickX-256;
int SpeedY=2*joystickY-256;
```

```
Serial.print("Speed: ");  
Serial.print(SpeedX);  
Serial.print(" ");  
Serial.println(SpeedY);
```

Also the serial would print out the speed. If unnecessary, you may add "//" at the beginning to remove it.

Now let's control the robot and make it move.

```
If ( move the Joystick up or (||) down ){  
    Robot move forward or backwards;  
}  
If ( move the Joystick to right or (||) left){  
    Robot move to the right or to the left  
}  
If ( X-axis of the Joystick keep in the center and (&&) Y-axis  
of the Joystick keep in the center ){  
    Robot stops;  
}
```

Correspondent codes are as follows:

```
if (SpeedX>200 || SpeedX<-200){  
    LeftWheelSpeed=SpeedX;  
    RightWheelSpeed=SpeedX;  
    Robot.Speed (LeftWheelSpeed,RightWheelSpeed);  
}  
else if (SpeedY>200 || SpeedY<-200){  
    LeftWheelSpeed=SpeedY-80;  
    RightWheelSpeed=-SpeedY-80;  
    Robot.Speed(LeftWheelSpeed,RightWheelSpeed);  
}  
else if (SpeedX==0 && SpeedY==0){  
    Robot.Speed(0,0);  
}
```

The last code is used to control the LED on the board connected to the PIN13.

```
int ledPin = 13;
```

Pin 13 should be define at the beginning of the code.

```
pinMode(ledPin, OUTPUT);
```

Set up output mode for LED by writing the function of setup(). Please refer to instruction for the function of pinMode() on the Arduino website([www.arduino.cc](http://www.arduino.cc)).

The following expressions show the state of the buttons. Once button No.1 has been pressed, LED would be on. Meaning that the LED pin is set to be HIGH.

```
if (buttonState[1] == PRESSED) {  
    digitalWrite(ledPin, HIGH);  
}
```

Once button No.1 has been released, LED would be turn off. Meaning that the LED pin is set to be LOW.

```
if (buttonState[1] == RELEASED) {  
    digitalWrite(ledPin, LOW);  
}
```

That's all for coding today. Isn't it FUN? SO, it's not impossible to write code for you, right? You may try to change a button to control the LED light by modifying the code. It's definitely more interesting when you can control it with your code. Now, **START HAVING FUN** with your robot!!!