

How to Control LEDs and Buzzers with LightBlue

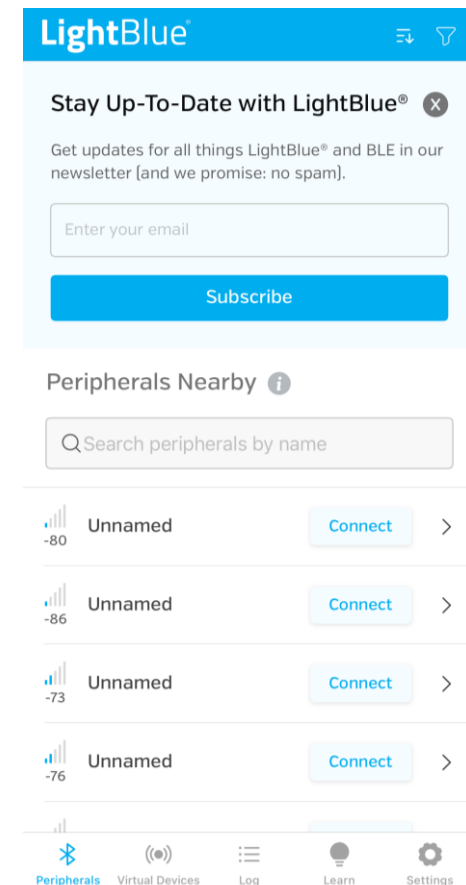
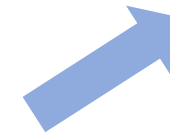
Software Used

Please install the following applications on your smartphone.

<https://punchthrough.com/lightblue/>

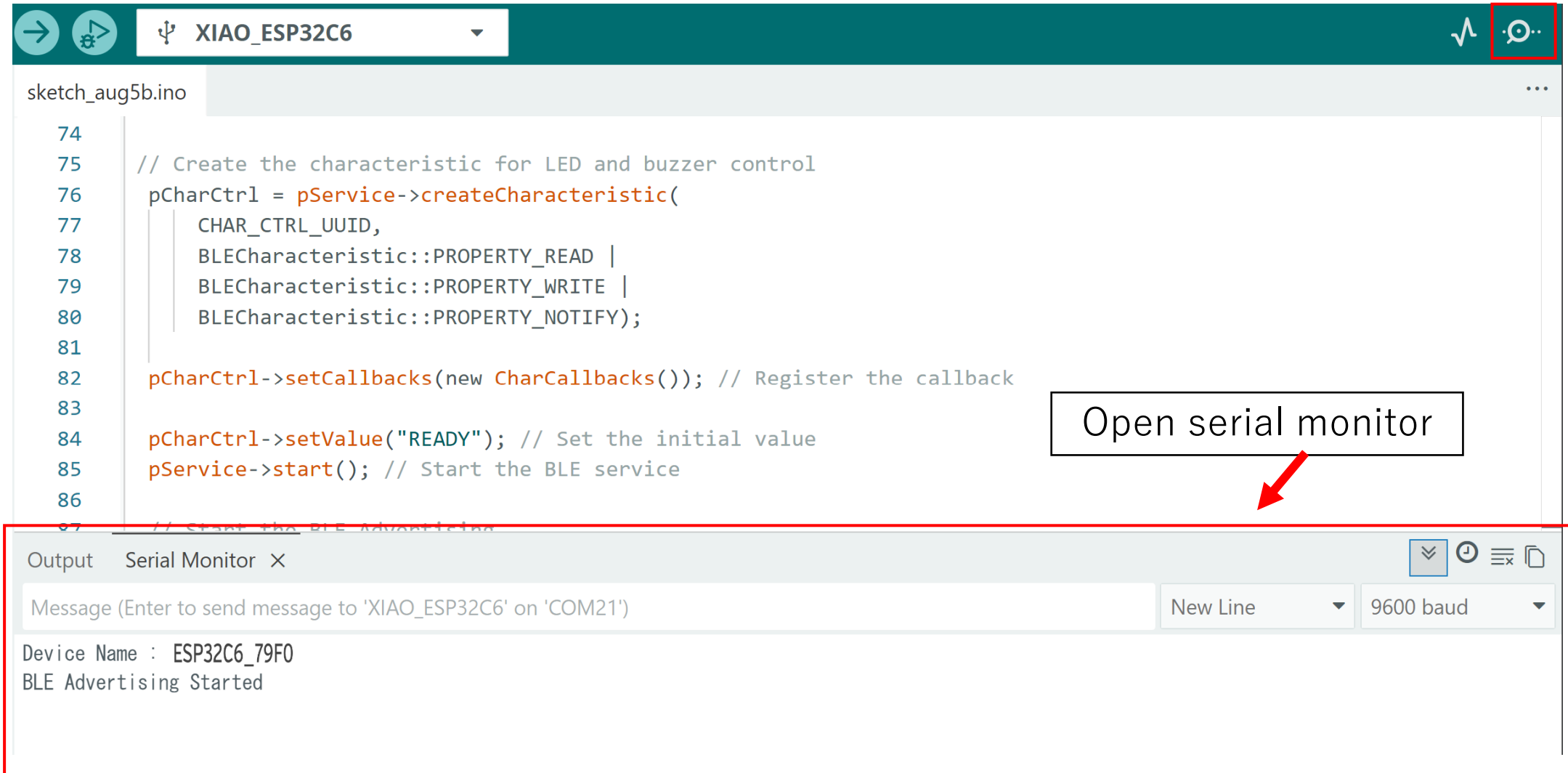


If this screen appears,
you're good to go.



How to Open Serial Monitor

Click here



The screenshot shows the Arduino IDE interface. At the top, the board is set to 'XIAO_ESP32C6'. The main editor displays a sketch named 'sketch_aug5b.ino' with the following code:

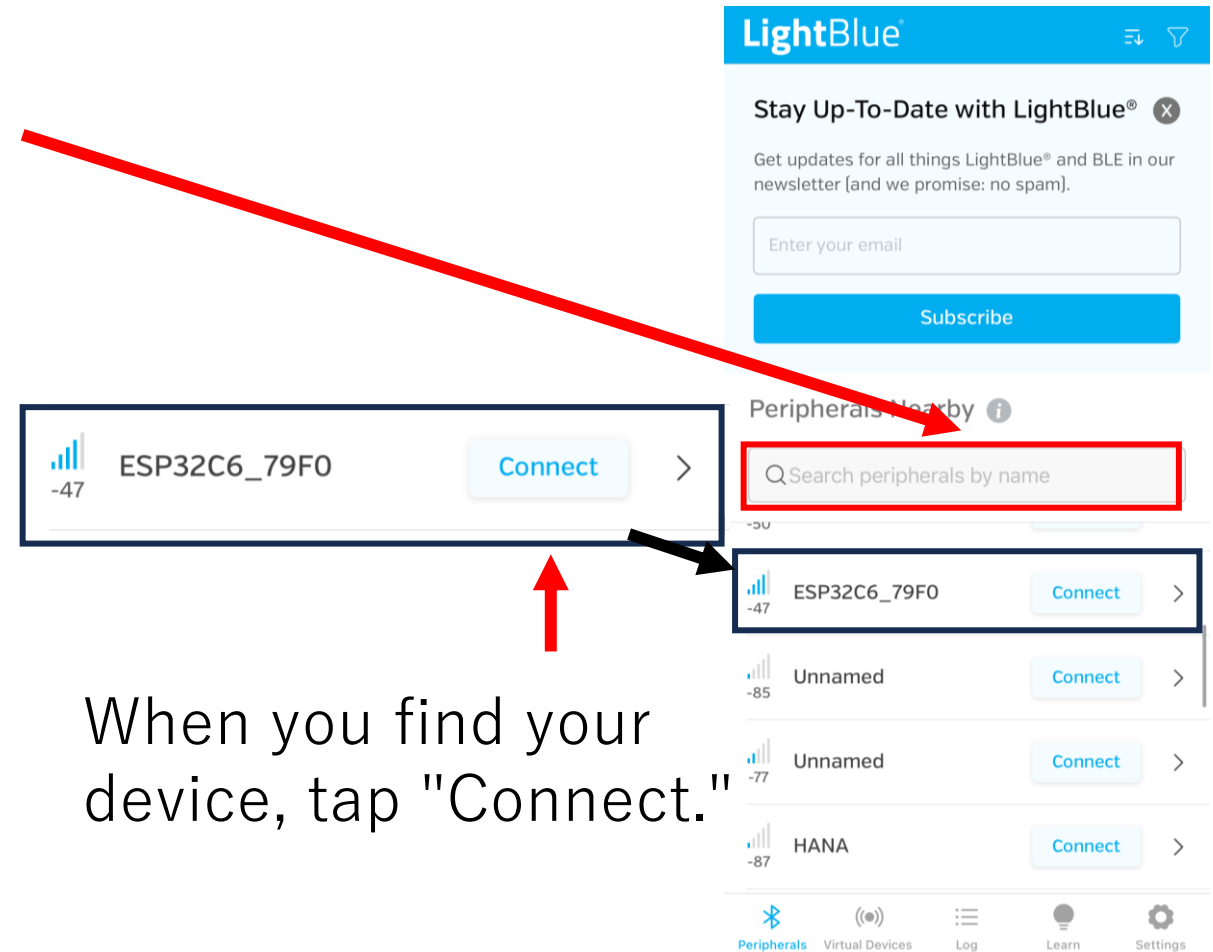
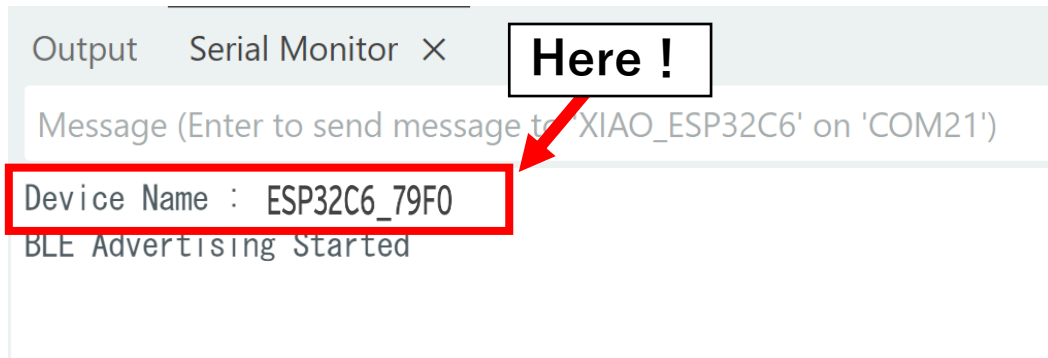
```
74
75 // Create the characteristic for LED and buzzer control
76 pCharCtrl = pService->createCharacteristic(
77     CHAR_CTRL_UUID,
78     BLECharacteristic::PROPERTY_READ |
79     BLECharacteristic::PROPERTY_WRITE |
80     BLECharacteristic::PROPERTY_NOTIFY);
81
82 pCharCtrl->setCallbacks(new CharCallbacks()); // Register the callback
83
84 pCharCtrl->setValue("READY"); // Set the initial value
85 pService->start(); // Start the BLE service
86
87 // Start the BLE Advertising
```

Below the code editor, the 'Serial Monitor' window is open. It shows the device name 'ESP32C6_79F0' and the message 'BLE Advertising Started'. The window also includes a text input field for sending messages, a 'New Line' dropdown, and a '9600 baud' dropdown.

Annotations in the image include a red box around the Serial Monitor icon in the top right of the IDE, with an arrow pointing to it from the 'Click here' text. Another red box is around the Serial Monitor window itself, with an arrow pointing to it from the 'Open serial monitor' text.

How to Use the App

- ① Use the Serial Monitor to check the name of your device.
- ② Find your device name.
(Searching is also available.)



Bluetooth Control of a Microcontroller !

Use a Bluetooth connection to send commands to the microcontroller.

① Click here.

②

Hex

③

④

Change the display settings. Click "Hex" in the upper-right corner, select "UTF-8 String", and then click "Save".

The image displays four sequential screenshots of a mobile application interface, illustrating the steps to change Bluetooth display settings. The first screenshot shows the 'Peripheral' screen with a list of services. The second screenshot shows the 'Characteristic' screen with the 'Hex' button highlighted. The third screenshot shows the 'Format Selection' screen with 'UTF-8 String' selected. The fourth screenshot shows the 'Format Selection' screen with the 'Save' button highlighted.

Screenshot 1: Peripheral

- Back
- Peripheral
- ESP32C6_B9E4
- UUID: 381478EF-3801-F3BE-FA6E-4BA9595310D0
- Connected
- Advertisement Data
- Services
- 0x44697E95-CE46-4FAC-BDF4-4BEE74427B06
- 0x9A155B2C-115A-4B3D-B36C-2DCC9A3143A9
- Properties: Notify, Read, Write

Screenshot 2: Characteristic

- Peripheral
- Characteristic
- Hex
- 0x9A155B2C-115A-4B3D-B36C-2DCC9A3143A9
- Connected
- Device: ESP32C6_79F0
- Service UUID: 44697E95-CE46-4FAC-BDF4-4BEE74427B06
- Write
- Write new value
- Written values will appear here
- Read/Notified values
- Read
- Subscribe
- Cloud Connect: OFF
- No value read recently
- Tap on one of the buttons above to begin

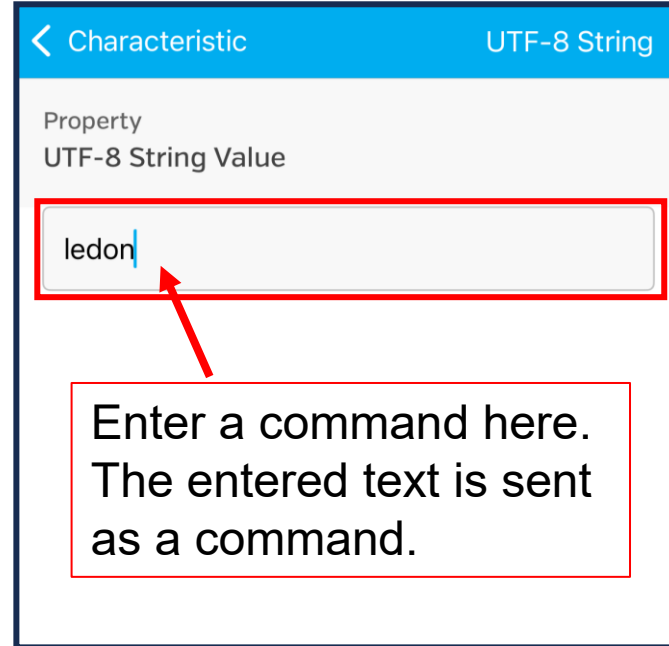
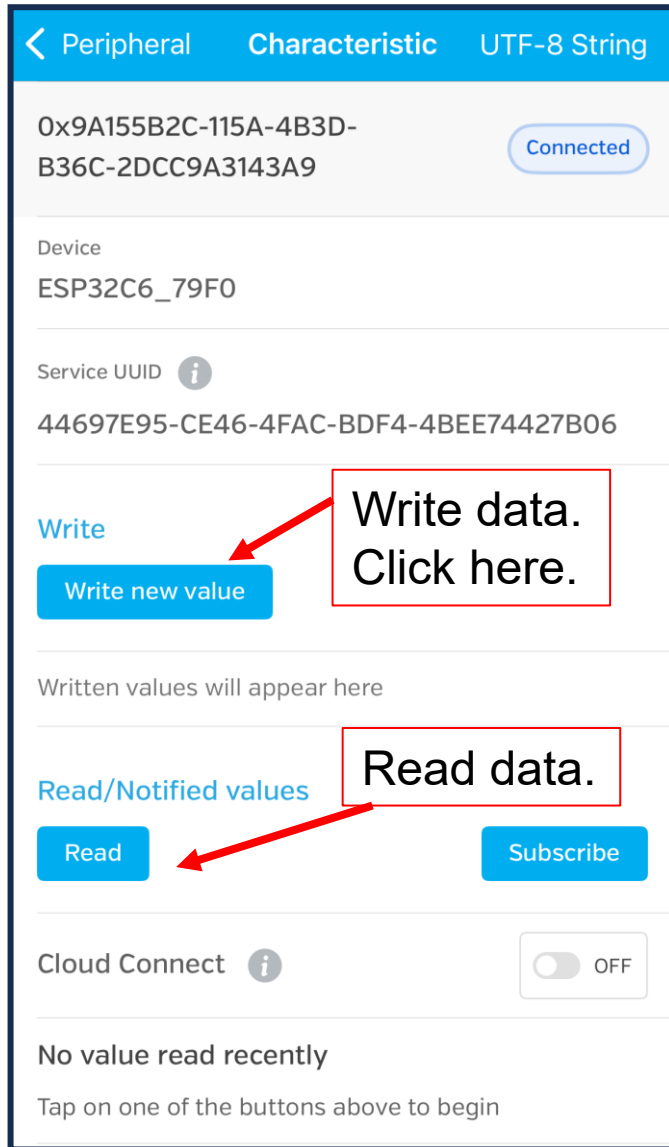
Screenshot 3: Format Selection

- Back
- Format Selection
- Save
- UTF-8 String
- No Value
- Binary
- No Value
- Hex
- No Value
- Octal
- No Value

Screenshot 4: Format Selection

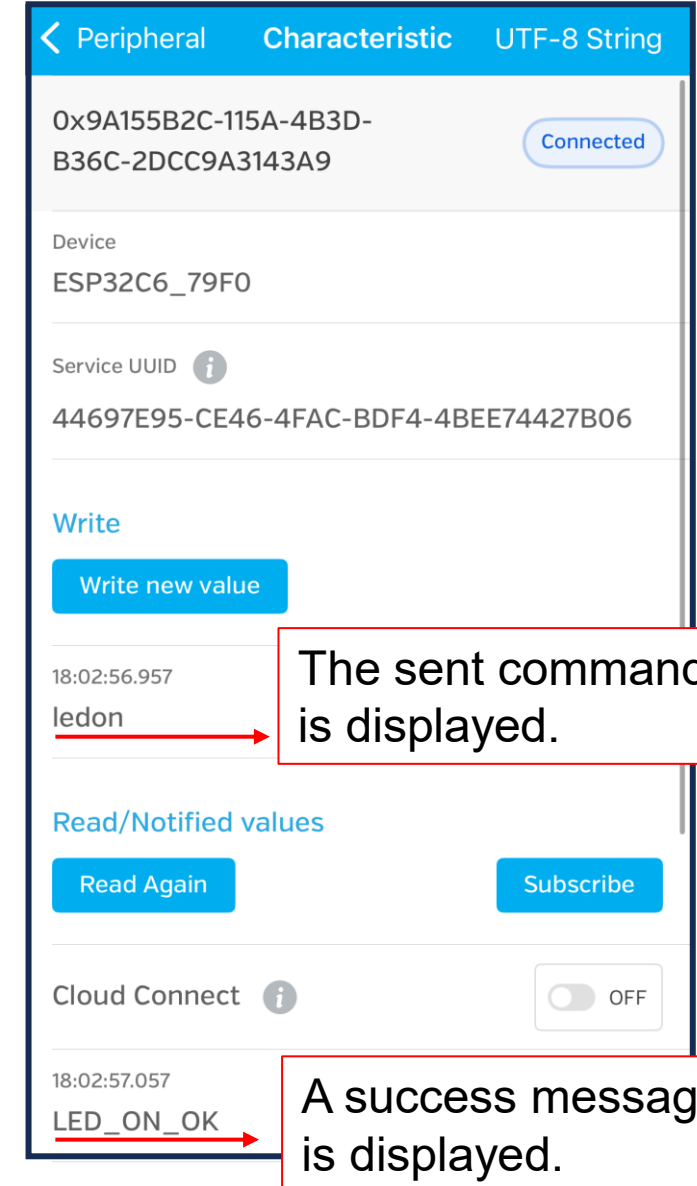
- Back
- Format Selection
- Save
- UTF-8 String
- No Value
- Binary
- No Value
- Hex
- No Value
- Octal
- No Value

Bluetooth Control of a Microcontroller !



Command	Action
ledon	Turn on the LED
ledoff	Turn off the LED
bz	Sound the buzzer

Uppercase and lowercase letters are both accepted.



Program Details(Overall Structure)

- **Settings and Variables** [**#define / Global Variables**]

Configure the LED and buzzer pin numbers and communication UUIDs.

- **Receive Action Definition** [**class CharCallbacks**]

Define the actions to perform when data is received from the smartphone (e.g., turn on the LED or activate the buzzer).

- **Initialization** [**void setup()**]

Runs once at startup. After creating the device name, it initializes the BLE server, service, and characteristic, registers the data reception callback, and starts BLE advertising to enter the standby state.

Program Details(Callback Function)

The callback function (CharCallbacks) is automatically executed whenever data is received from a BLE device, such as a smartphone.

```
class CharCallbacks : public BLECharacteristicCallbacks {  
    void onWrite(BLECharacteristic *pChar) override {  
        String value = pChar->getValue();  
        value.toLowerCase();  
        Serial.print("Received: ");  
        Serial.println(value);  
        if (value == "ledon") {  
            digitalWrite(LED_PIN, HIGH);  
            // Notify the execution result  
            pChar->setValue("LED_ON_OK");  
            pChar->notify();  
        }  
    }  
};
```

←“onWrite” is called when data is received.

←Read the received data as a string.

←Accept both uppercase and lowercase commands.

←Display the command on the Serial Monitor.

←If the command is “ledon”, turn the LED on (HIGH).

←Send “LED_ON_OK” to the smartphone.

←Notify the smartphone that the command was executed.



Command received!
Run the callback function!
Turn on the LED!