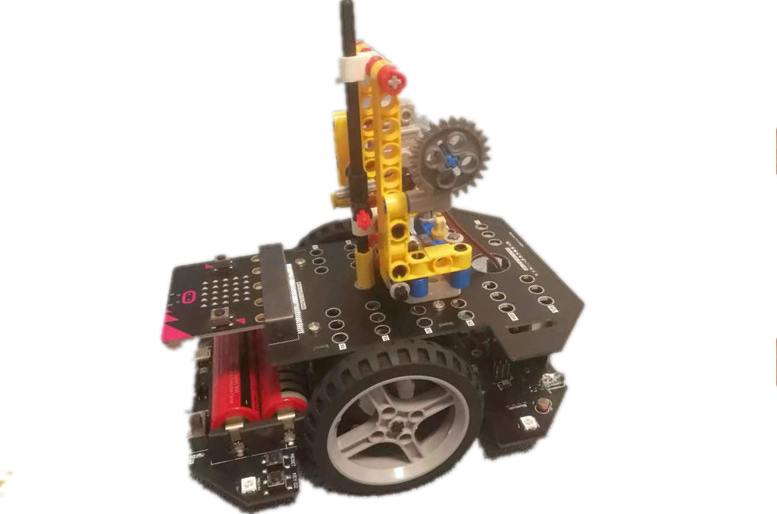
# Overview



Cobit 是一款围绕着创客教育与DIY而设计的机器人小车，使用步进电机驱动，可以进行精准距离行走。集成了超声波模块、光敏传感器、led与RGB灯、无缘蜂鸣器、红外接收传感器、红外巡线传感器，可外接蓝牙模块和ESP-01 wifi模块。兼容arduino、microbit、esp wifi，可以使用C、C++、JavaScript、Python、Blocks语言进行编程，车体上有乐高安装孔和IO口，可以扩展为写字机器人，也非常方便外接各类型模块，从7+岁的儿童到大学生和电子爱好者都合适使用。

## Support

☛质量问题

☛使用疑惑

☛学习与技术交流

☛优化方案

☛创意交流

请发邮件到✉：[jex-spt@outlook.com](mailto:jex-spt@outlook.com)

本邮箱在工作日24小时为您服务，为了更好的为您服务，请在提出问题时清晰说明产品的SKU，并尽可能地详细描您得问题。

## Colophon

Copyright © 2021 Cokoino Intelligent Technology Co., Ltd..

build-date: 2021-08-03

build-version: V1.0

Web: http://cokoino.com

## Legal Disclaimer Notice

Cobit已进行相关专利申请，任何个人或者团体组织仿造或者盗版相关技术，cokoino公司有权追究其相关法律责任。

本产品涉及的代码和电路被发布github上：[https://github.com/cokoino/CKK0008](https://github.com/cokoino)，您可以在您自己的派生作品中部分或全部使用它们，只要您也采用它们相同的许可证。Cokoino品牌及logo版权归Cokoino创作团队所有未经正式许可不得用于任何商业用途。

## **Table of Contents**

# **Tech specs**

Power source: Two lithium batteries.

Speed rating:

0 level: 0 revolutions per minute.

1 level: 15 revolutions per minute.

2 level: 30 revolutions per minute.

3 level: 60 revolutions per minute.

4 level: 120 revolutions per minute.

Wheel diameter: 65mm.

Walking precision: 0.51mm.

Precision of rotation angle: 0.9°.

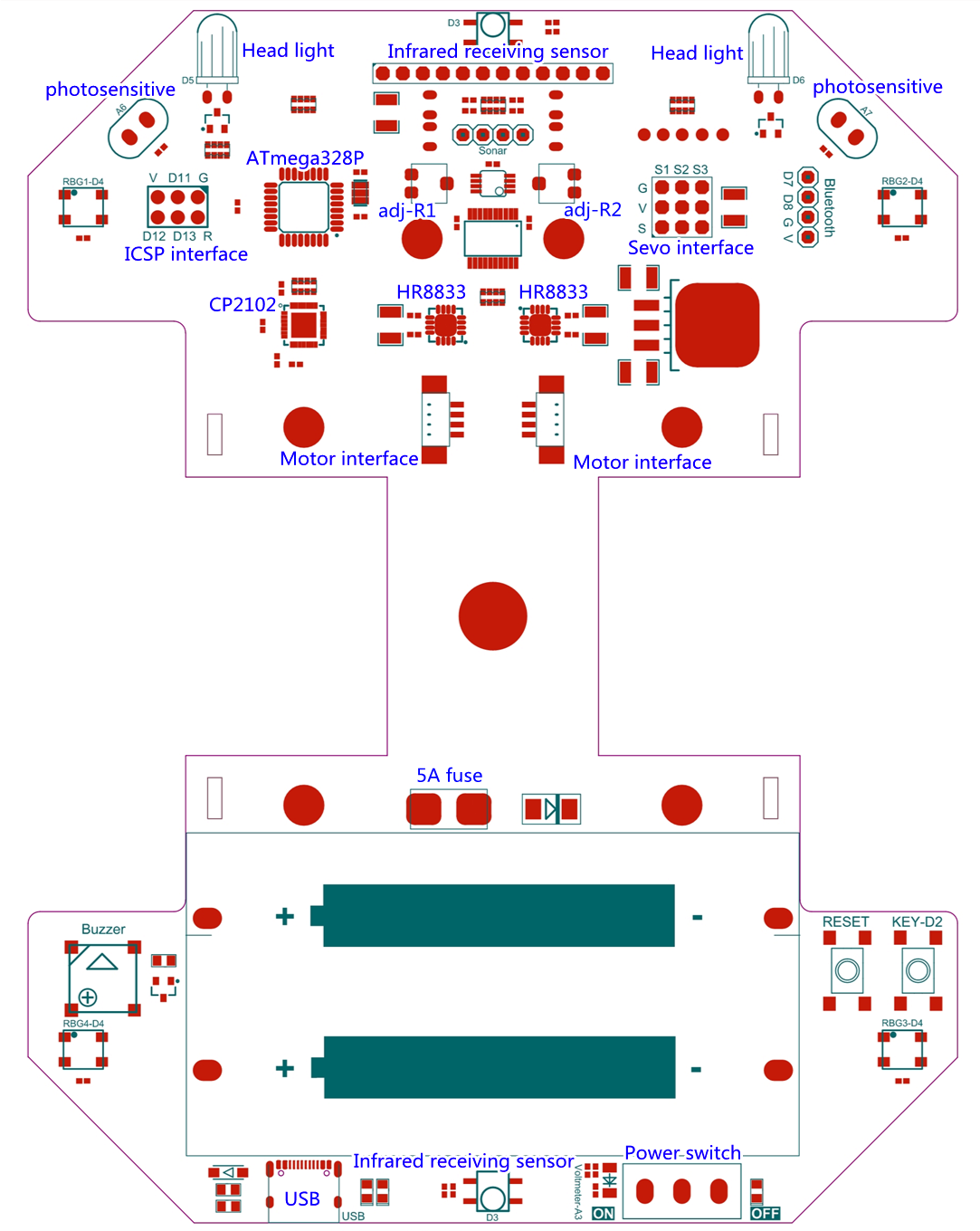
Drive power: 110mN.m REF. per [stepper](D:/Youdao/Dict/8.9.6.0/resultui/html/index.html" \l "/javascript:;) motor.

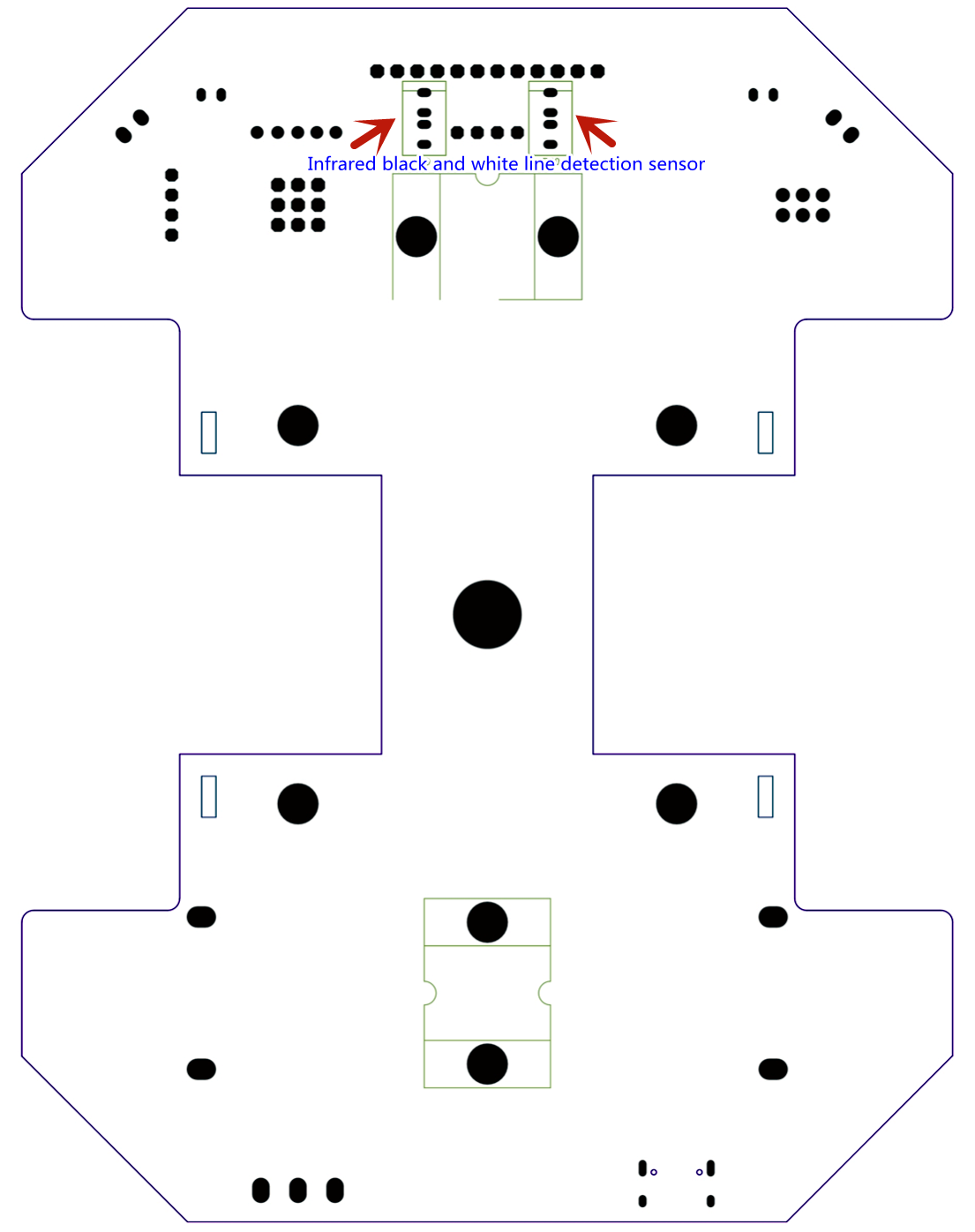
Buzzer can be set frequency range: 20-- 1K Hz, volume: 0-5 level.

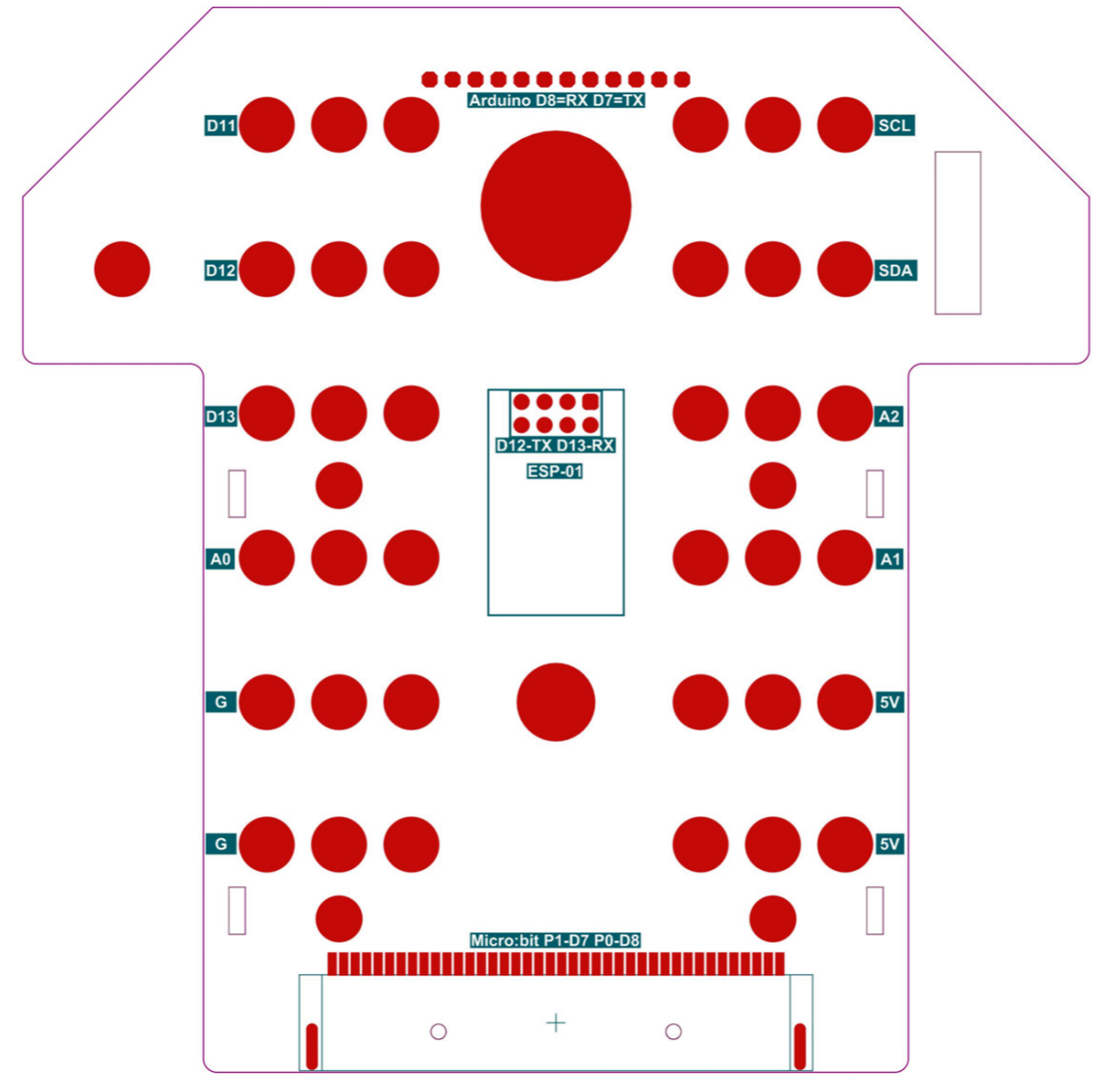
Maximum measurement distance of sonar is 255CM with an accuracy of +/-1CM.

# **Hareware**

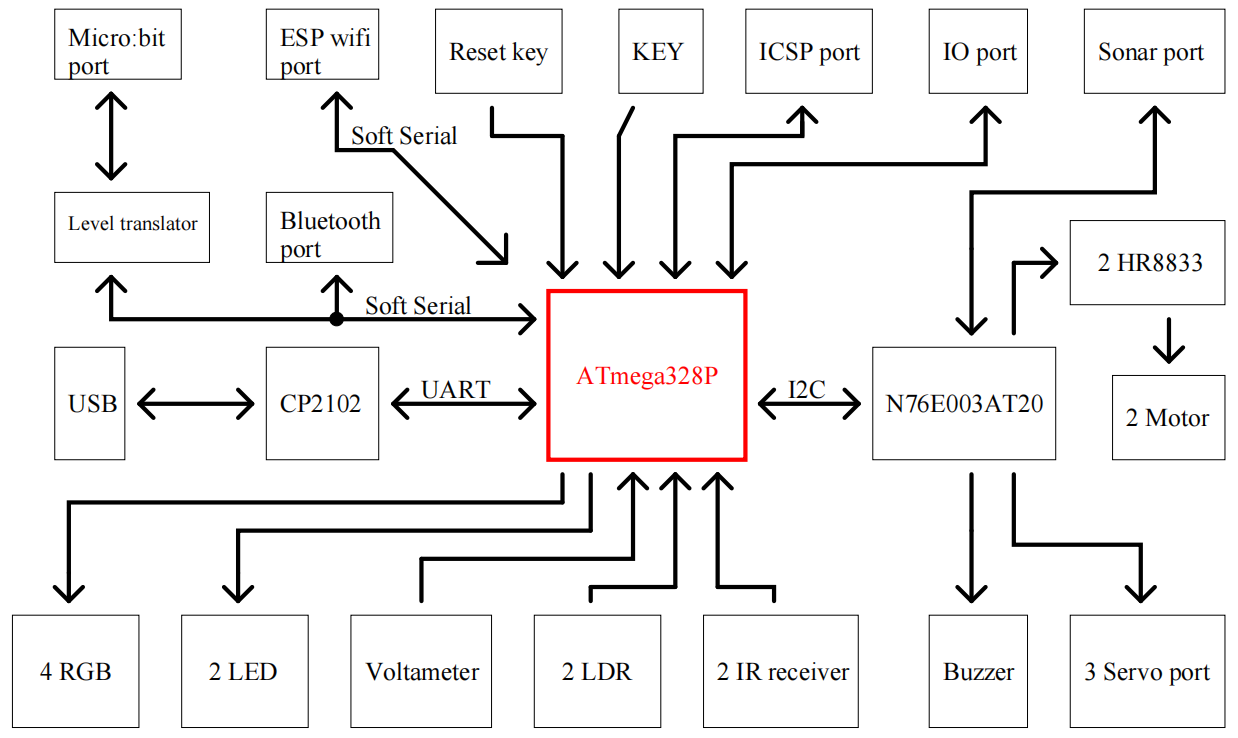
## Overview







## **Hardware block diagram**



## **Hardware Description**

**ATmega328P：主控芯片**

核心控制器，可以通过USB口更新代码。

|  |  |
| --- | --- |
| Flash Memory | 32 KB of which 0.5 KB used by bootloader |
| SRAM | 2 KB |
| EEPROM | 1 KB |
| Clock Speed | 16 MHz |
| Bootloader | Arduino UNO R3 |

**N76E003AT20：辅助芯片**

烧录了特定的代码，通过I2C协议与ATmega328P通信，用于驱动DRV8833RTYR、Servo、Sonar与蜂鸣器。

**CP2102：USB to serial chip**

将USB信号转换为TTL Serial电平信号。

|  |  |
| --- | --- |
| Max Baud Rate | 921600 |
| USB protocol | USB Specification 2.0 compliant; full-speed (12 Mbps) |
| Driver Support | Windows 10/8/7/Vista/XP/Server 2003/2000 Windows CE |

**HR8833：电机驱动芯片**

双H桥电机驱动器，可用于驱动2个直流电机或1个4线2相步进电机，受控于N76E003AT20的内部程序，ATmega328P的I2C端口发送相关命令给N76E003AT20可以控制它。

|  |  |
| --- | --- |
| Power Voltage Range: | 2.7 V – 15 V |
| Output Current | 1.5-A RMS per H-Bridge |
| MOSFET On-Resistance: | HS + LS 400 mΩ |

**电机接口：**

用于连接4线2相步进电机，分为左右两个，受控于HR8833芯片，具体参数请看上叙述。

**红外接收传感器：**

中心频率38K赫兹，接收距离5-10米，默认使用NEC红外通信协议，信号脚连接与ATmega328P的D3脚。

**无源蜂鸣器：**

最大可以产生1K赫兹的声音，受控于N76E003AT20的内部程序，ATmega328P的I2C端口发送相关命令给N76E003AT20可以控制它。

**红外黑白巡线传感器：**

通过红外反射原理来感知黑白线，分为左右两个，信号脚分别连接于ATmega328P的D9与D10脚，当遇到白线时红外形成反射，可以获取到反射回来的信号，信号脚输出低电平；当遇到黑线时，红外线被吸收，没有反射信号，信号脚输出高电平。通过adj-R1与adj-R2可以调节传感器的灵敏度，出厂已调节，非必要情况不建议再次调节。

**光敏传感器：**

读取光的强度，分为左右两个，输出模拟值，信号脚分别连接于ATmega328P的A6与A7脚，光线越强光敏电阻的阻值越小，内部电路输出的电压值越大。

**RGB灯：**

WS2812 RGB灯，可以发出可控的红、绿、蓝三原色光，总共4个灯，序号分为RGB1-RGB4，通过分别控制三原色的强度可以形成各种颜色的光，信号输入脚连接与ATmega328P的D4脚。

**Head light：**

5mm白色LED灯，分为左右两个，分别受控于ATmega328P的D5与D6脚，通过D5、D6脚的PWM信号可以使LED灯发出不同强度的白光。

**Type C USB口：**

USB通信接口，通过它可以将PC上的代码、数据等发送给cobit，也可以使用它更新ATmega328P的程序。

**电源开关：**

控制电池盒电源的开启与关闭，电池盒线路有一个5A保险与一个防止防接二极管，更为安全。

**电压计：**

用于检测电池盒的电压，输出模拟值，信号脚连接与ATmega328P的A3脚。因为A3得到的为1/10分压后的电池电压，所以A3得到的数值要乘以10才为真正的电池电压。

**KEY-D2按键：**

信号脚连接与ATmega328P的D2脚，按键按下输出低电平，否则输出高电平。

**RESET按键：**

此信号脚连接于ATmega328P的复位脚，按键按下时触发ATmega328P复位。

**超声波接口：**

V=5V，G=GND，用于外接HC-SR04 sonar模块，受控于N76E003AT20的内部程序，ATmega328P的I2C端口发送相关命令给N76E003AT20可以读取HC-SR04数据。当外接HC-SR04 sonar模块时，最大测试距离为255CM，精度位+/-1CM。

**舵机接口：**

V=5V，G=GND，S=舵机控制信号，用于外接3路舵机，受控于N76E003AT20的内部程序，ATmega328P的I2C端口发送相关命令给N76E003AT20可以控制外接的舵机。

**蓝牙接口：**

用于外接蓝牙模块，V=5V，G=GND，信号脚直接连接于ATmega328P的D8（连接于蓝牙模块的TX引脚）与D7（连接于蓝牙模块的RX引脚）引脚。

**ICSP接口：**

V=5V，G=GND，主要用于烧录ATmega328P的bootloader。

**ESP-01 wifi接口：**

用于外接ESP-01 wifi模块，TX与RX信号脚通过3V-5V电平转换电路连接于ATmega328P的D12（对应esp-01的TX引脚）与D13（对应esp-01的RX引脚）引脚。

**Micro:biti接口：**

用于外接Micro:bit主板，P0与P1信号脚通过3V-5V电平转换电路连接于ATmega328P的D8（对应Micro:biti的P0引脚）与D7（对应Micro:biti的P1引脚）引脚。

**数字口：D11、D12、D13**

为ATmega328P的数字IO口，其中D11可以输出PWM信号。

**模拟口：A0、A1、A2**

为ATmega328P的模拟口，可以读取模拟值，也可以当作数字口使用。

**电源输出接口：G、5V**

GND = G，电源 = 5V，最大输出功率为10W，可用于外围设备供电。

**I2C口：SCL、SDA**

为ATmega328P的I2C通信接口，SDA=A4，SCL=A5，可与其他I2C设备进行通信。

## **Further information**

I2C Protocol：

<https://www.nxp.com/docs/en/user-guide/UM10204.pdf>

NEC Infrared Transmission Protocol：

<https://techdocs.altium.com/display/FPGA/NEC+Infrared+Transmission+Protocol>

# **Install the cobit**

# **Geting start with cobit for arduino**

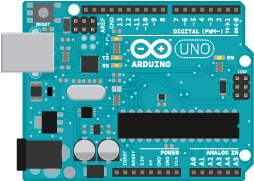
Arduino is an open-source electronics platform based on easy-to-use hardware and software. You

can use the Arduino programming language (based on Wiring), and the Arduino Software (IDE),

based on Processing.

官网链接：<https://www.arduino.cc/>

## **Arduino 开发板**



Arduino 是一个开源平台，官方开发出许多开源的开发板供电子爱好者制作与使用，其中比较著名的为Arduino UNO R3，它是一款基于ATmega328P控制芯片的主板，cobit的ATmega328P芯片也是使用基于它的bootloader。

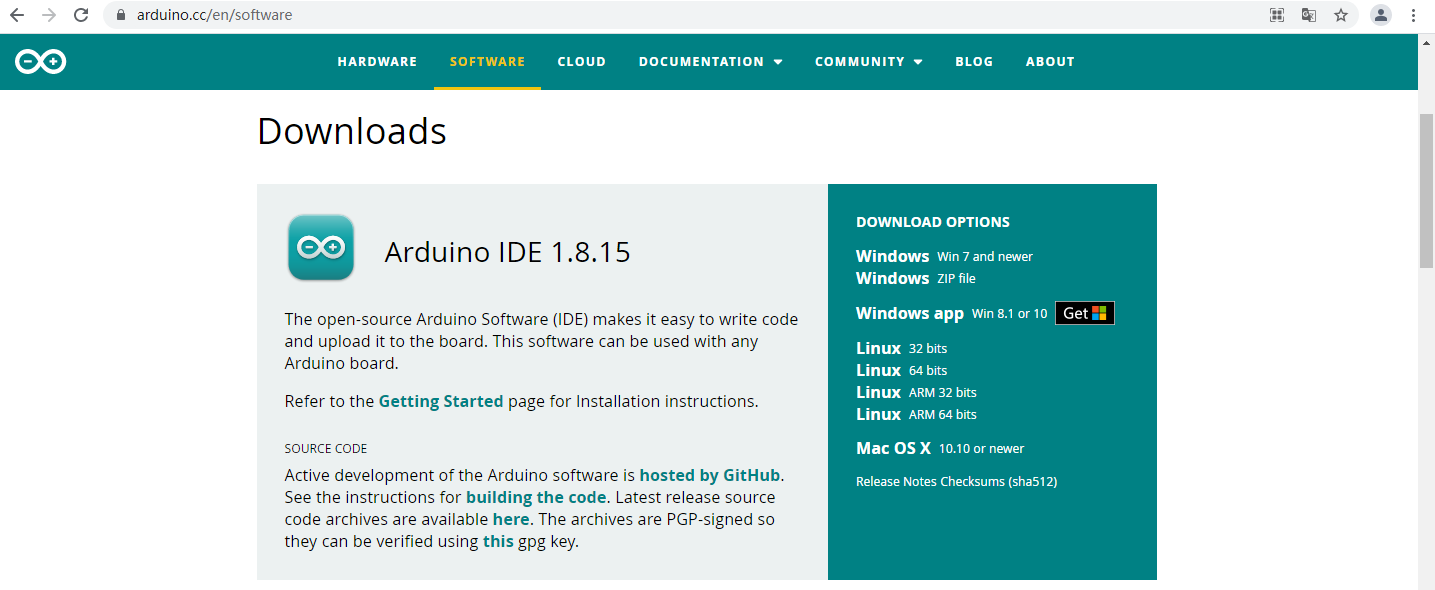
Arduino 开发板链接：

<https://www.arduino.cc/en/Main/Products>

## **Arduino IDE**

开源的Arduino软件(IDE)使编写代码和上传代码变得容易。主要是要的是C与C++编程语言，预留有丰富的API接口，可以满足各类开发需求。

**安装arduino IDE**

进入arduino官网 [https://www.arduino.cc/，](https://www.arduino.cc/%EF%BC%8C) 点击  .  


选择合适你PC系统的安装包，这里我们以windows系统示例。  
  
点击进入安装包下载界面。



点击开始下载安arduino安装包，安装包下载完成后，请解压安装包进行安装。

更多安装细节与不同PC系统的安装方法也可以参考：<https://www.arduino.cc/en/Guide>



**Arduino IDE菜单**

IDE VERIFY File：Verify Checks your code for errors compiling it.

IDE UPLOAD File：Upload Compiles your code and uploads it to the configured board. See [uploading](https://www.arduino.cc/en/Guide/Environment" \l "uploading) below for details.

Note: If you are using an external programmer with your board, you can hold down the "shift" key on your computer when using this icon. The text will change to "Upload using Programmer".

IDE NEW File：New Creates a new sketch.

IDE OPEN File：Open Presents a menu of all the sketches in your sketchbook. Clicking one will open it within the current window overwriting its content.

Note: due to a bug in Java, this menu doesn't scroll; if you need to open a sketch late in the list, use the File | Sketchbook menu instead.

IDE SAVE File：Save Saves your sketch.

IDE SERMON File：Serial Monitor Opens the [serial monitor](https://www.arduino.cc/en/Guide/Environment" \l "serialmonitor).

更多IDE使用详细资料请参考：<https://www.arduino.cc/en/Guide/Environment>

## **Arduino 编程语言**

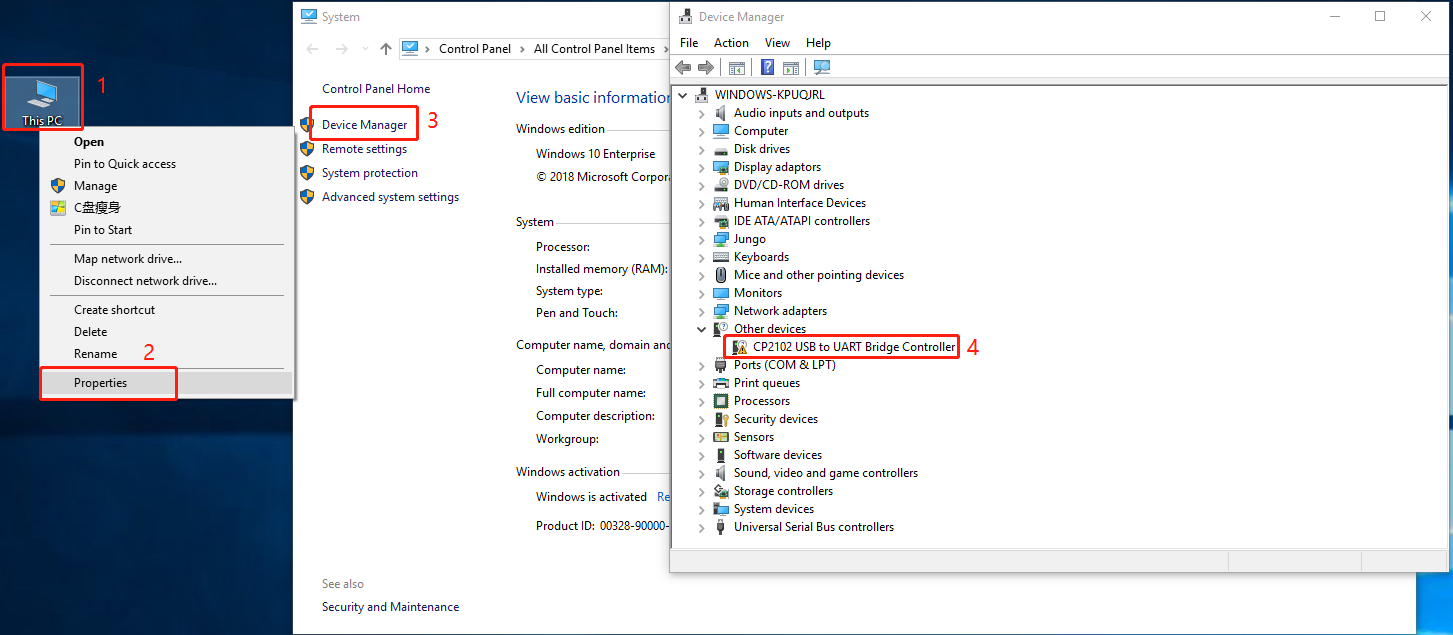
Arduino有一套基于C与C++的编程语法与API接口，可以实现各种功能，Arduino编程语言可以分为三个主要部分:函数、值(变量和常量)和结构。

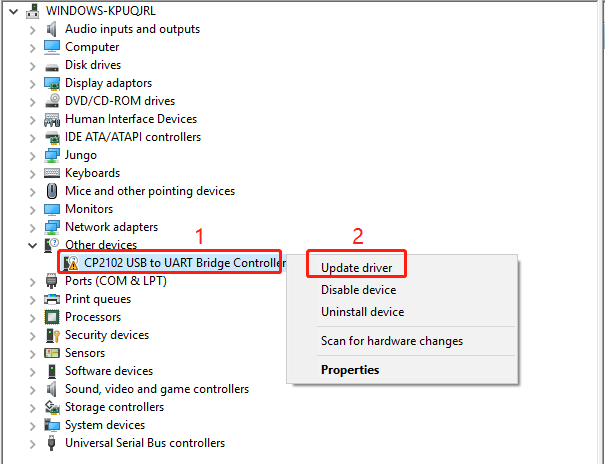
具体学习资料请参考：<https://www.arduino.cc/reference/en/>

## **Install the USB-to-UART COM port driver**

Cobit上集成了一个Type-C USB口，并集成了一颗CP2102 USB -to-serial芯片，使用它可以与ATmega328P的串口进行通信，使用前必须在PC上安装CP2102 驱动程序。

Arduino 1.8以上的版本的IDE已将CP201X的驱动程序包含到内部了，在连接PC的USB口时，PC会自动识别并安装驱动程序。

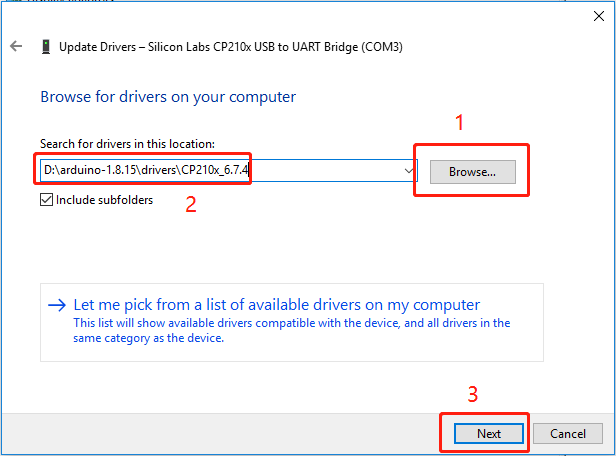
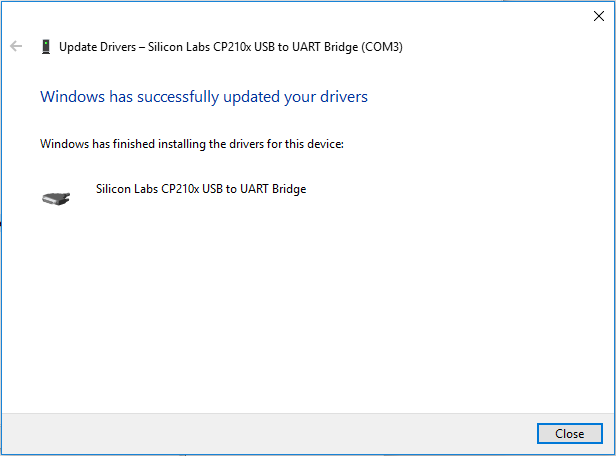
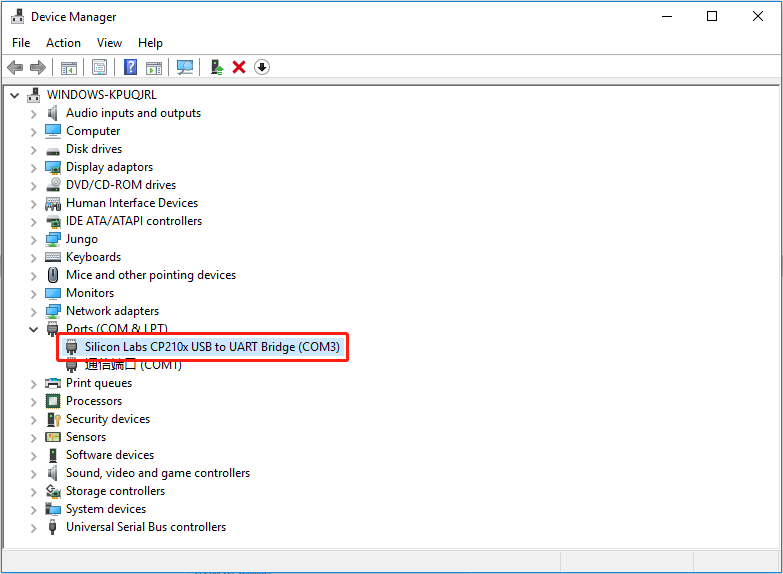
如果驱动安装失败或者想手动更新驱动，在PC桌面选中”This PC”菜单，然后点击鼠标右键选择”Properties”--->” Device Manager”。  


这设备管理器页面有一个黄色叹号的设备，如上图。选中此设备，然后点击鼠标右键，再选择”Update driver”。  


点击“browse my computer for updated driver software”。  

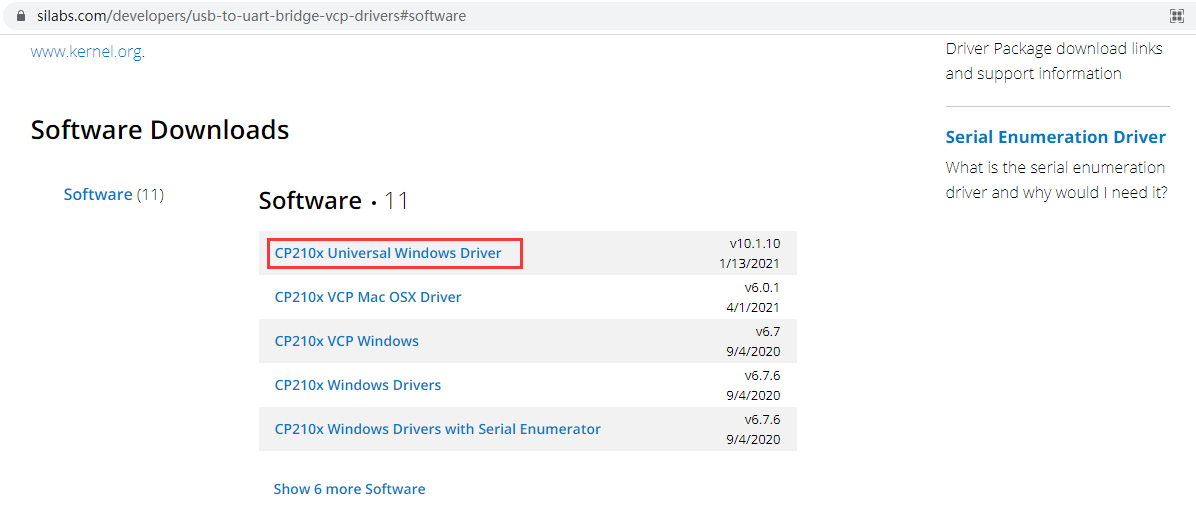

There is a DRIVERS folder in Arduino software installed package（）, open driver folder and you can see the driver of CP210X series chips.

We click “Browse”, then find out the driver folder, or you could enter “driver” to search in rectangular box, then click “next”, the driver will be installed successfully.

  
  
Open device manager, we will find the yellow exclamation mark disappear. The driver of CP2102 is installed successfully.  
  


驱动也可以在原始官网下载：

<https://www.silabs.com/developers/usb-to-uart-bridge-vcp-drivers#software>

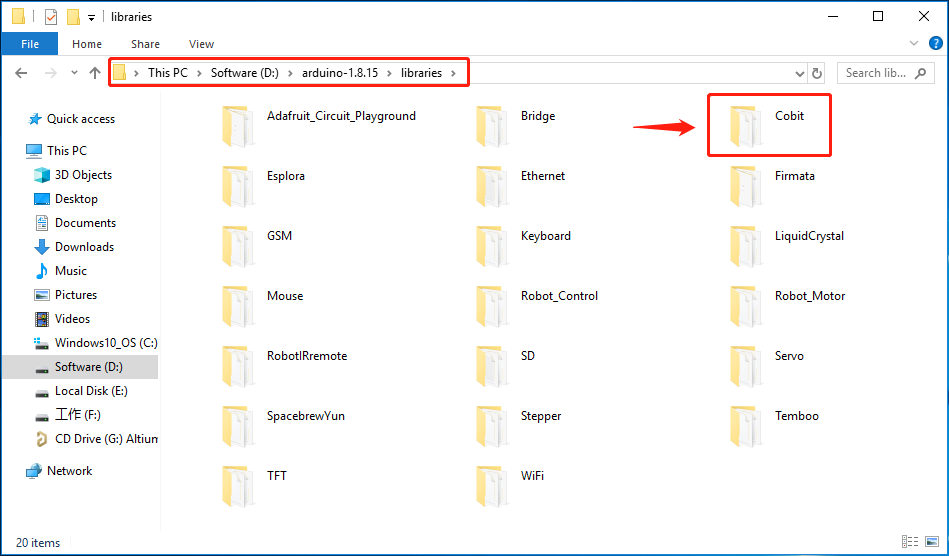


## **Cobit使用arduino平台**

Cobit使用了一颗ATMEGA328P作为核心处理器，在出厂时烧录了Arduino UNO R3的引导程序，因此它是完全兼容Arduino UNO R3，在arduino平台上使用时完全按照UNO R3的参数设置即可。

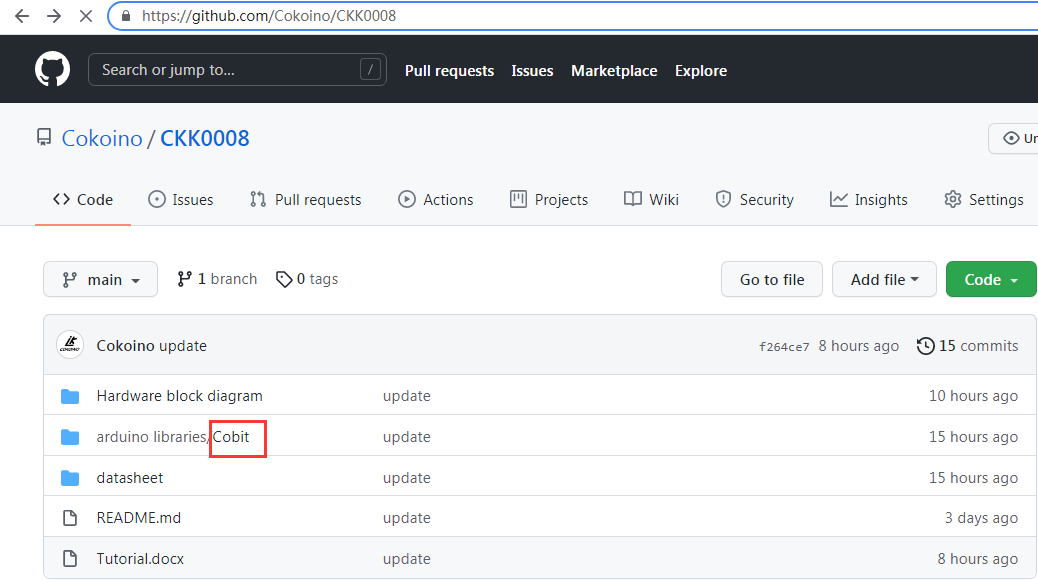
## **Arduino library for cobit**

我们使用arduino 平台为cobit开发了一个库文件，此库包含了cobit的所有功能，使得cobit使用arduino编程变得更为简单与高效，使用此库只需要将我们提供的或者github上下载的”cobit”压缩包解压到Arduino安装目录下的”libraries”文件夹中，如下图：



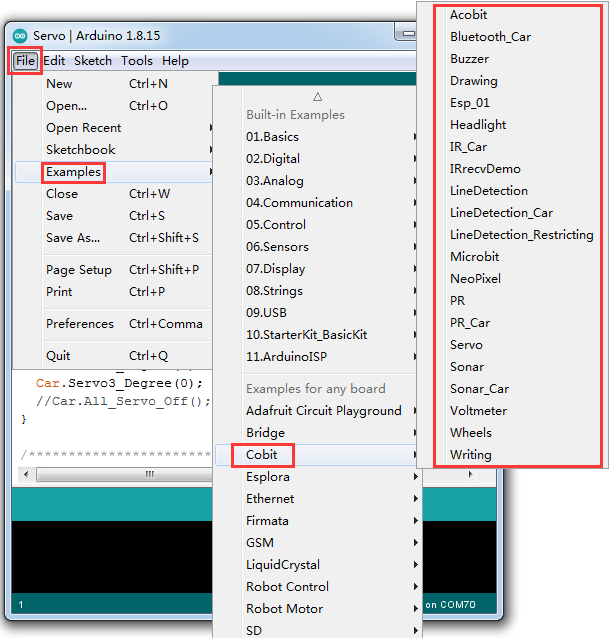
备注：此库基于UNO R3开发板。

Cobit库存放于：<https://github.com/Cokoino/CKK0008> 的arduino libraries文件夹中。



**Cobit的arduino示例：**

我们的所有功能都集成在arduino库里面，在上一步骤我们已经将库文件解压到arduino IDE的”libraries”文件夹里面，现在我们在arduino IDE的示例里面即可找到cobit的所有示例代码，如下图：



注意：将cobit库文件解压到Arduino安装目录下的”libraries”文件夹后，需要关闭IDE重新启动才能在IDE的示例中找到cobit的示例。

**一个简单示例**

选择代码：

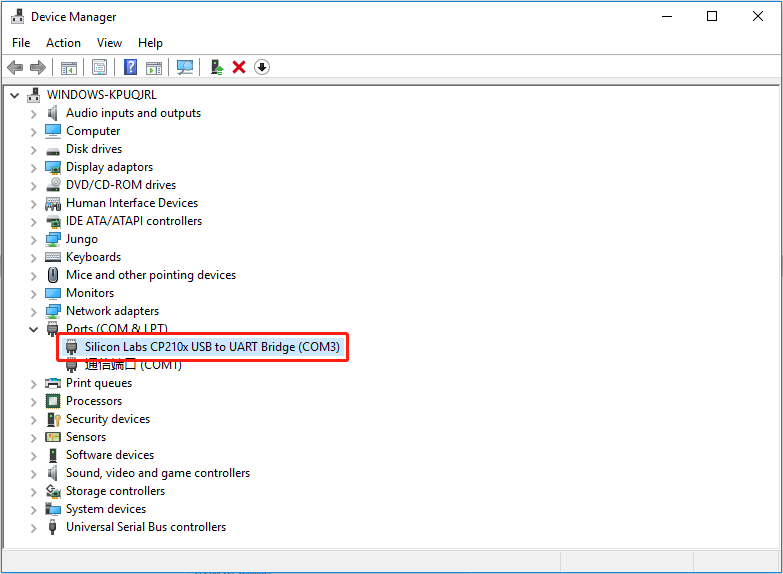


选择板型：

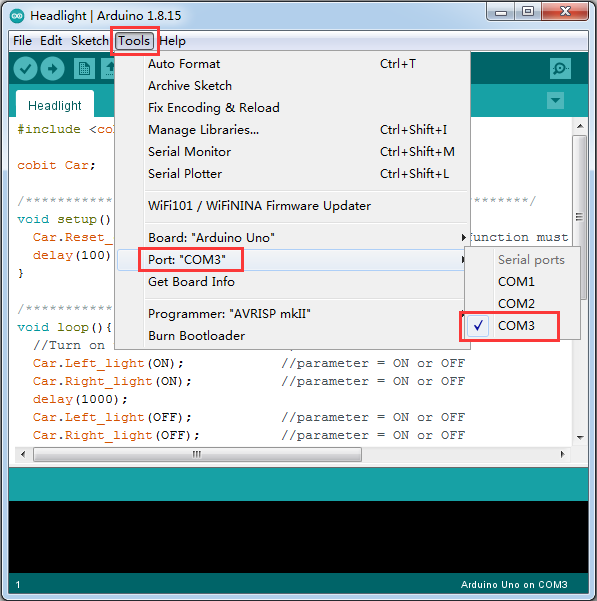


选择端口：

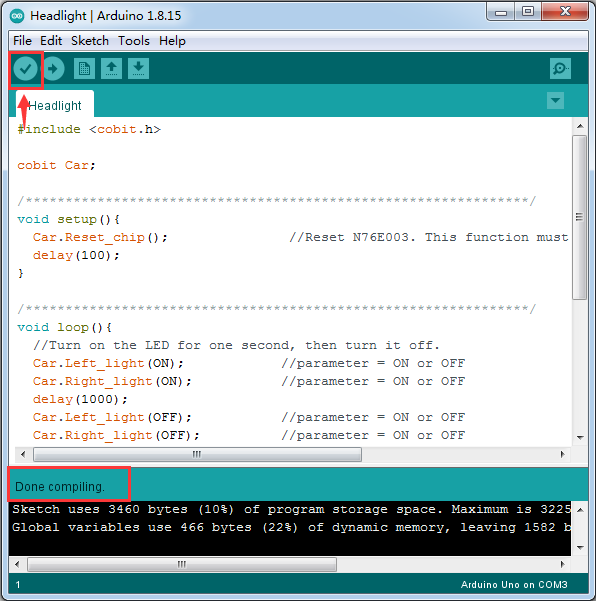
使用Type C USB线连接cobit与PC，打开PC的设备管理器（如果设备显示叹号，请参考”Install the USB-to-UART COM port driver”章节安装驱动）。



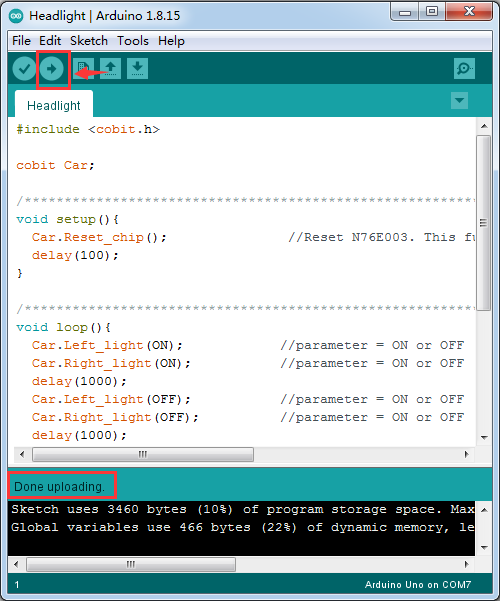
依据设备管理器的端口在arduino IDE中选择对应的端口。



校验代码，如果代码有错误，此步骤无法通过，IDE底部的信息窗口打印橙色错误信息。



上传代码：



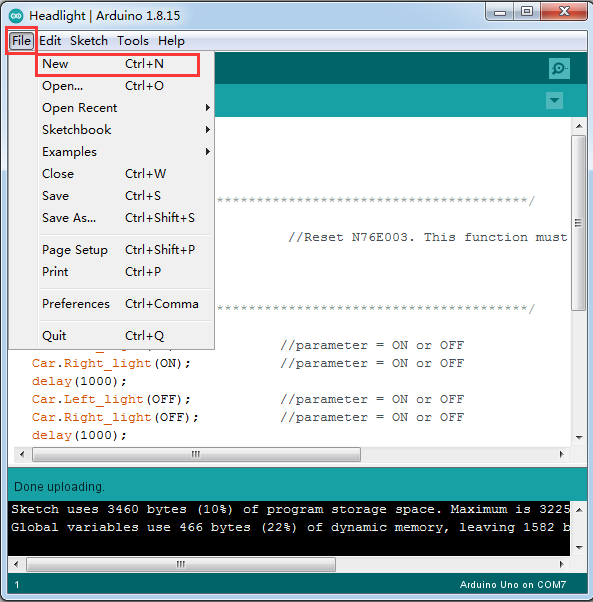
代码效果：

Cobit两个车头灯先点亮1秒，然后熄灭；接着会呼吸灯点亮，然后呼吸灯熄灭；一直如此循环。

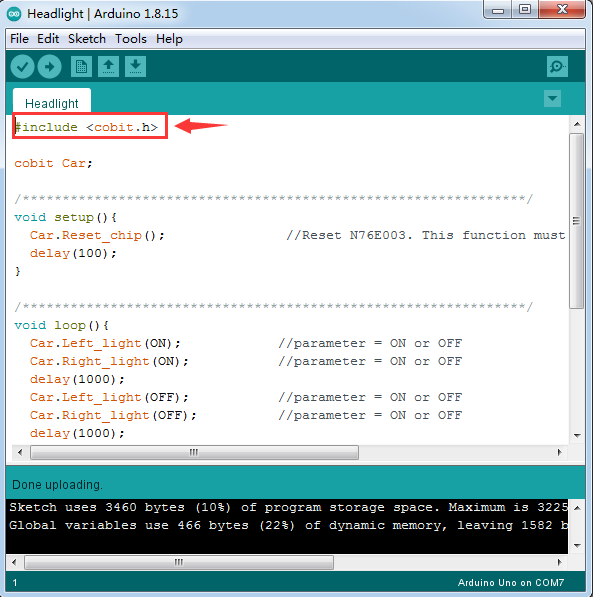
**新建cobit工程：**

如果想使用cobit实现自定义功能，只需重新新建一个工程并保存，在代码中包含必须的头文件，然后再使用示例预留的API接口或者参考库文件的解析选择所需的功能，如下图。

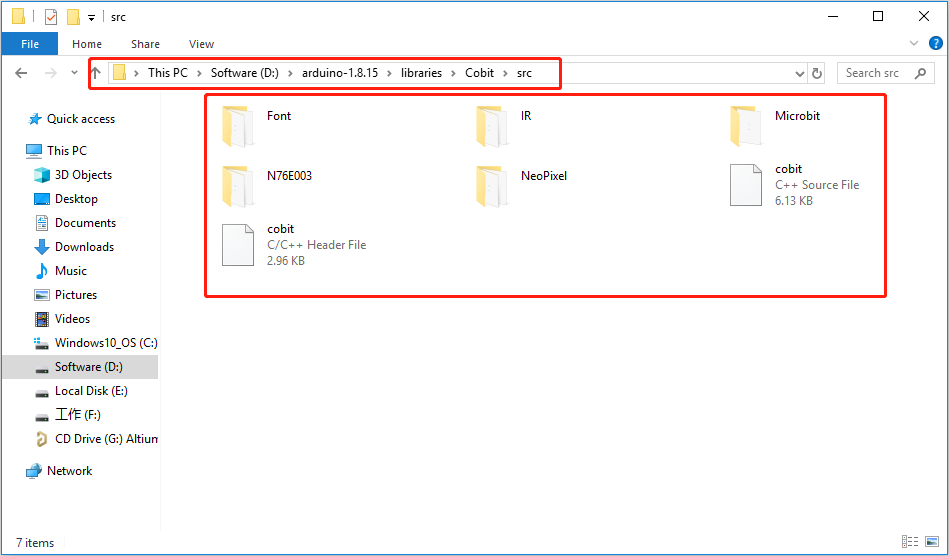
新建工程：



添加头文件：



查阅库文件的函数使用方法：





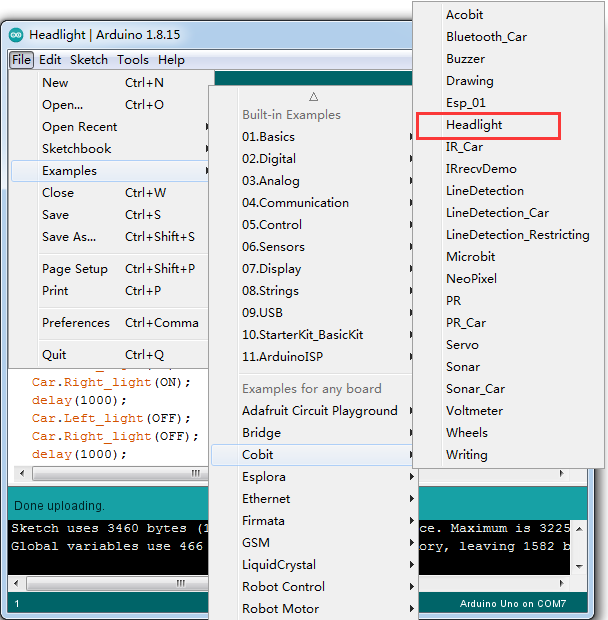
提醒：推荐使用”Notepad++”软件查阅上述文档，下载网址：

<https://notepad-plus-plus.org/downloads/>

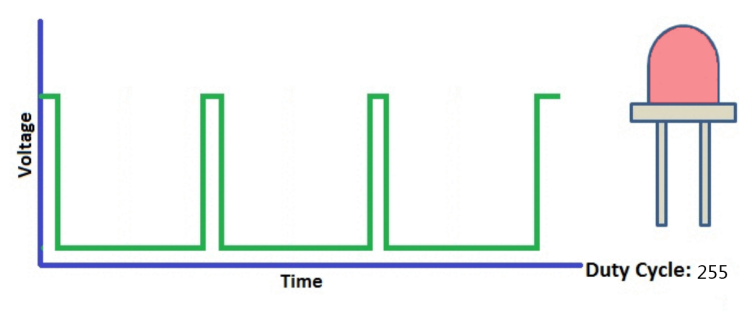
## **示例课程**

Cobit的示例课程都已集成的cobit库里面，因为这样非常方便又避免了拷贝导致的错误。使用示例时请认真查看代码中”//”后面的注释，这会非常有助于你理解代码的功能。

### 车头灯示例

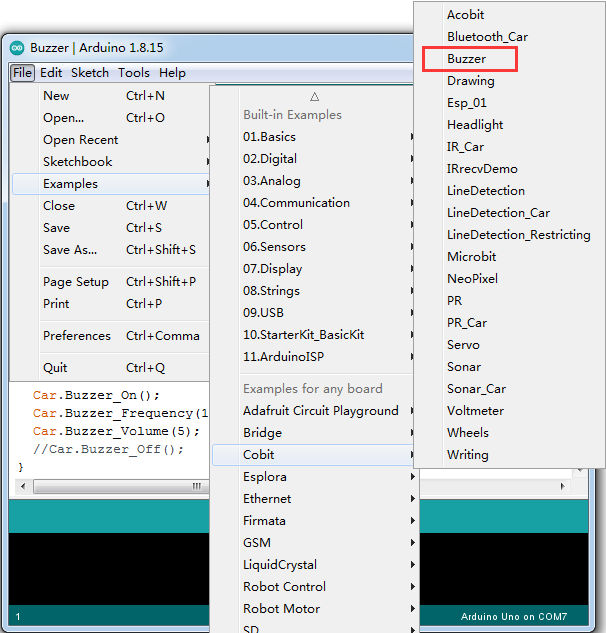


Cobit集成了两个白色5MM的车头灯，“Headlight”提供对它们开启与关闭两种状态的控制，还可以通过控制它们的亮度，从而实现呼吸灯的效果。

灯的亮度是通过PWM信号来控制，PWM为脉宽调制的意思，即周期性产生高低电平时间宽度可变的脉冲。当高电平的时车头灯会点亮，低电平时车头灯熄灭，例如一个脉冲的周期为255毫秒（周期=高电平时间+低电平时间），当高电平时间宽度为255毫秒时，低电平时间宽度为0，此时车头灯最亮；当高电平时间宽度为0，低电平的时间宽度为255毫秒时，此时车头灯最暗。

上传代码后，Cobit两个车头灯先点亮1秒，然后熄灭；接着会呼吸灯点亮，然后呼吸灯熄灭；一直如此循环。

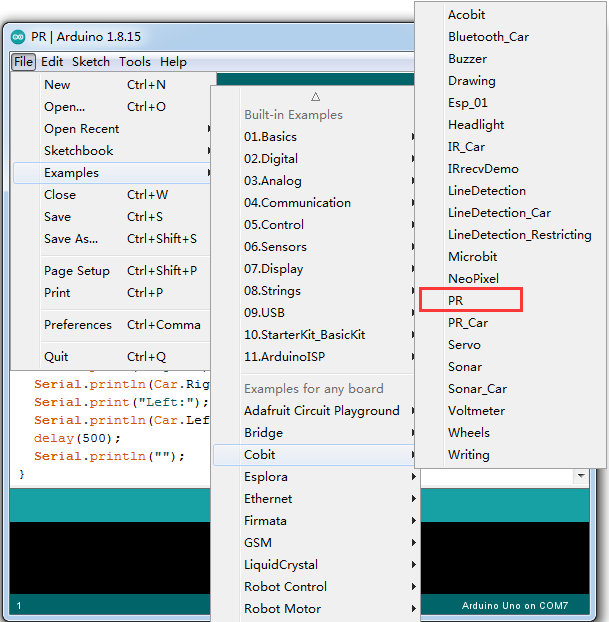
### 蜂鸣器示例



Cobit有一个无源蜂鸣器，当一直给无源蜂鸣器直流电时，蜂鸣器并不会响，有的还会导致蜂鸣器发烫，严重时还会导致蜂鸣器损坏，只有给无源蜂鸣器一个有频率的信号源才能产生响声。“Buzzer”代码提供了对无源蜂鸣器开启与关闭两种状态的控制，还可以调节它的音量与声音频率（20---1000Hz）。

上传代码后，蜂鸣器会发出20---1000赫兹的声音，一直如此循环。

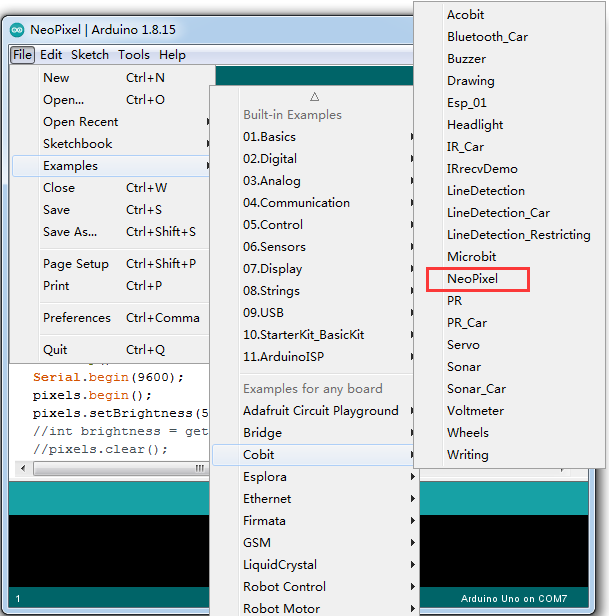
### 光敏电阻示例

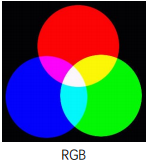


Cobit前端集成了两个光敏电阻，“PR”代码可以读出它们的模拟值。

上传代码后，打开arduino IDE的串口监视器，将波特率设置为9600，监视器将打印两个光敏电阻的数据。

### RGB灯示例

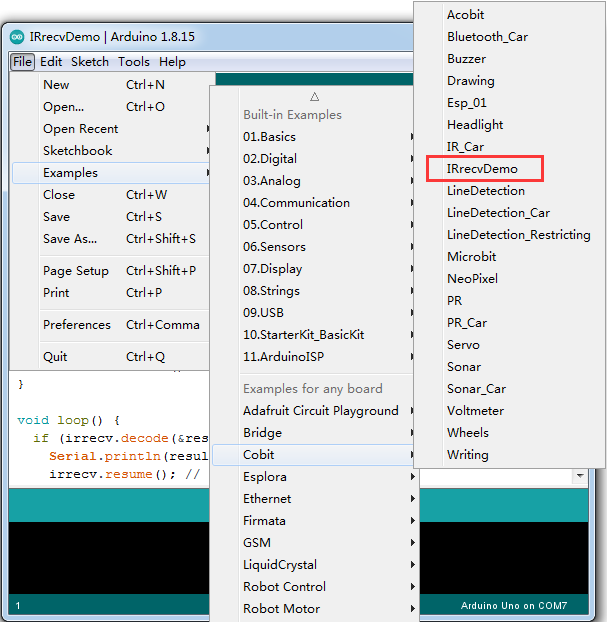




Cobit有4个RGB灯，可以发出红、绿、蓝三种光的灯，通过控制三种光的亮度，再相互组合既可以产生各种颜色的光。

上传代码后，将会流水点亮4个RGB灯（发白光），然后再流水熄灭。

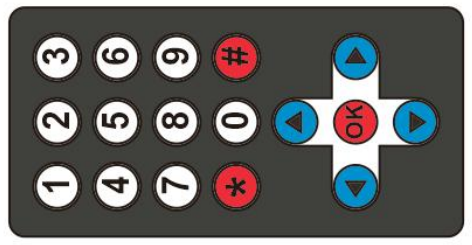
### 红外接收示例

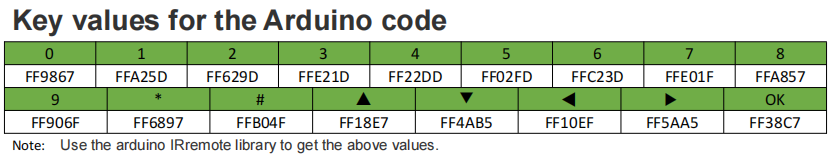


Cobit前后各有一个红外接收传感器，这样的设计使得cobit在接收数据时的死角更小，接收到的数据更准确，更加灵敏。

上传代码后，打开串口监视器，将波特率设置为9600，然后使用我们提供的红外遥控器或者兼容NEC协议的遥控器对着cobit发送按键数据，串口监视器将打印遥控器的按键值。

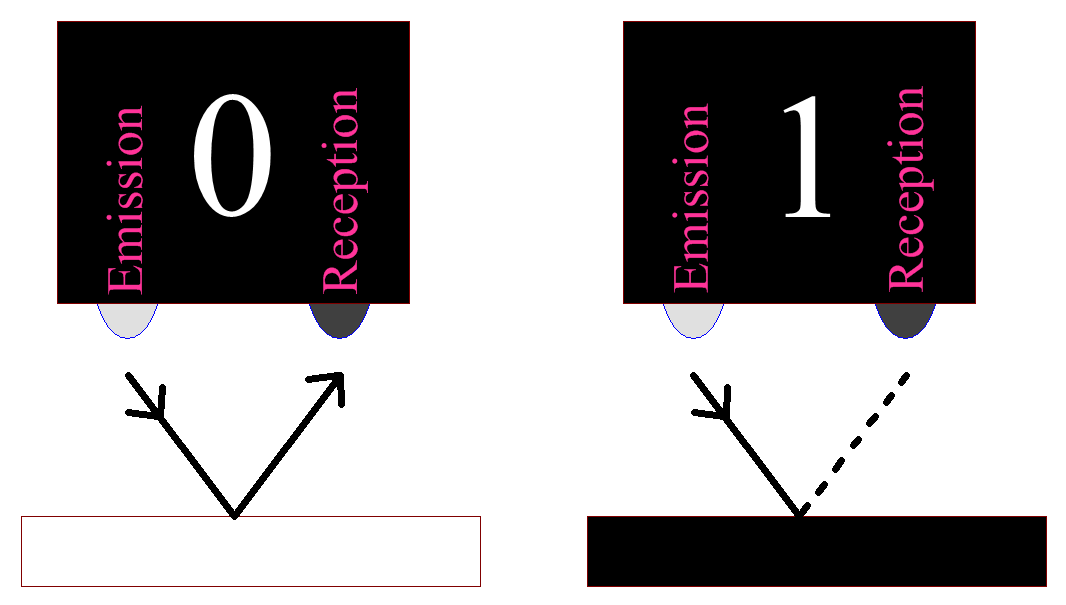
红外遥控器按键值：





### 红外黑白线识别示例

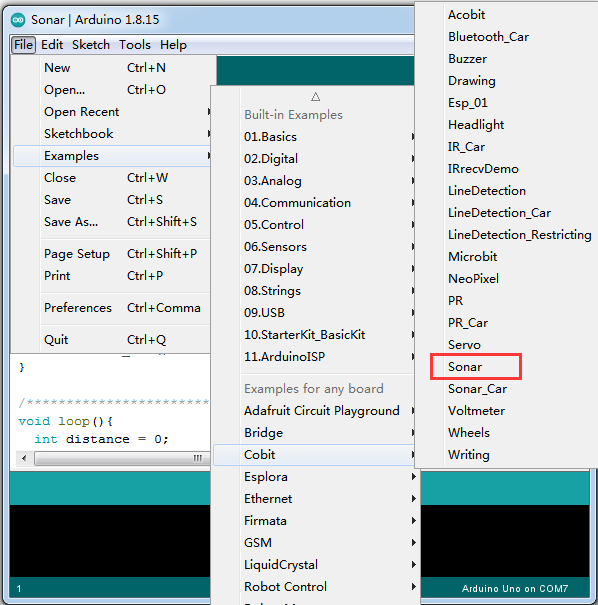


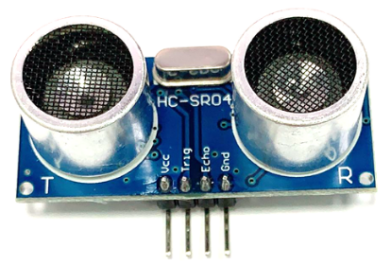
Cobit的底部集成了两个红外对管巡线传感器，初始默认输出高电平。红外对管分为两部分，一部分为红外发射端，一部分为红外接收端，红外发射端发射红外信号，接收端接收碰到物体后反射回来的红外线信号。

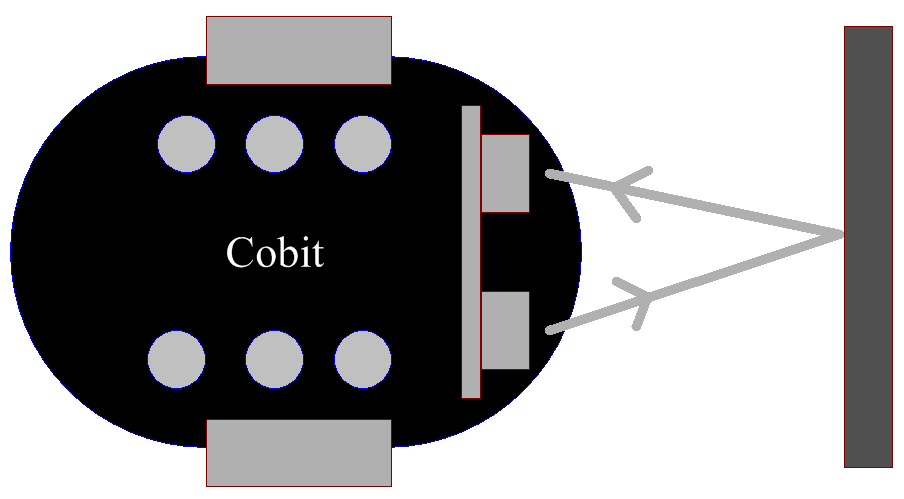
黑色线对红外线有很好吸收效果，当红外对管巡线传感器的发射端发射红外线照射到黑色线时，红外线被吸收没有形成反射信号，接收端没有接收到信号，巡线传感器输出高电平（1）；当照射到白色线时，因为白色线对红外线没有吸收效果，所以红外线信号被反射回来，接收端接收到信号，巡线传感器输出低电平（0）。

上传代码后，打开串口监视器，将波特率设置为9600，将cobit翻转过来，使用白色纸张靠近两个红外传感器，如果左侧红外传感器检测到串口监视器打印"left detected"，如果左侧红外传感器检测到串口监视器打印"right detected"，如果左右的红外传感器都检测到串口监视器打印"left detected"与"right detected"，如果左右的红外传感器都检测到串口监视器打印"left detected"与"right detected"。

### Sonar示例

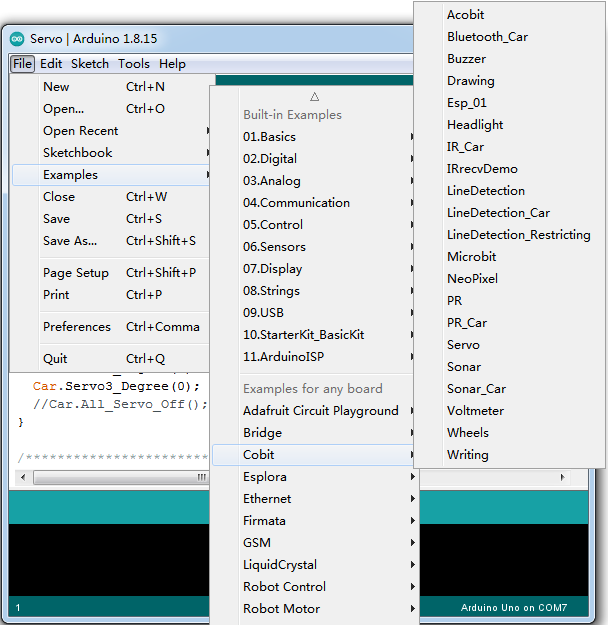


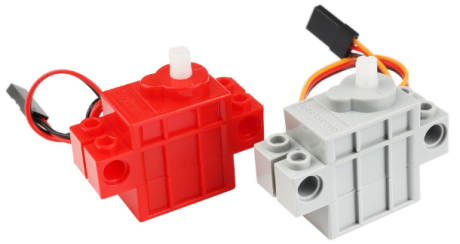


Cobit上预留有一个sonar接口，可以外接HC-RS04超声波模块进行测量距离。Cobit默认配给有HC-RS04超声波模块，并将模块的API集成在cobit库里面，使用API即可读取模块的测量数据，大大降低了使用难度。最大测量距离为255cm，精度为+/-1CM。

上传代码后，打开串口监视器，波特率设置为9600，将一个障碍物放在cobit正前方，串口会将cobit与障碍物之间的距离打印出来，数据单位为厘米。

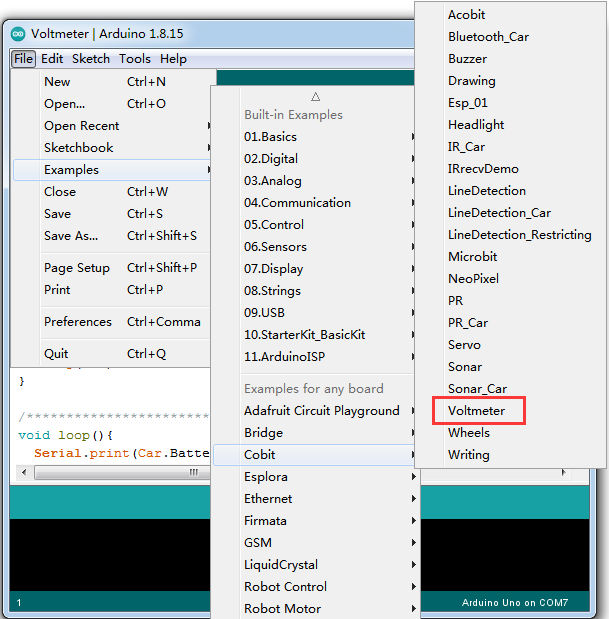
### Servo示例



Cobit上预留有3个servo驱动接口，可以外接乐高舵机或者SG90舵机。Cobit默认配给有一个乐高舵机，并将模块的API集成在cobit库里面，使用API即可简单快速驱动舵机，大大降低了使用难度。

上传代码后，将舵机接入任意一个舵机接口，舵机将在0---180度之间来回摆动。

### 电压计示例

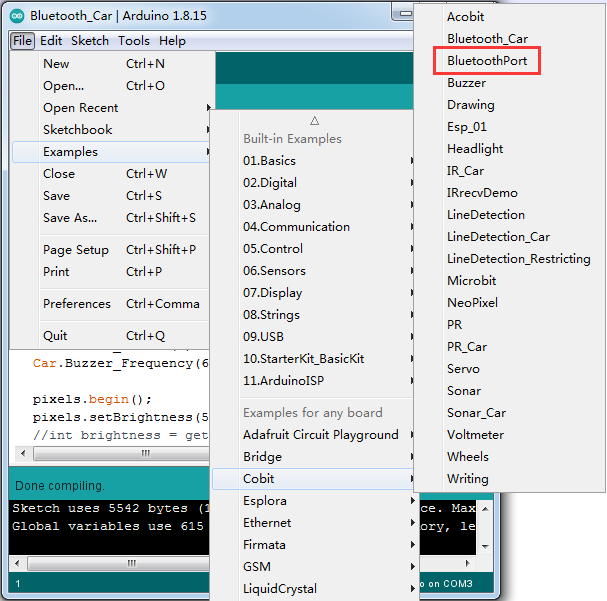


Cobit上设计了一个电压计用于读取电池的电压，使用Cobit库中的API可以快速地读取电池是电压和电压百分比，电压百分比是一个映射值：

|  |  |
| --- | --- |
| 电压范围 | 映射电压百分比 |
| 6.5V---8.4V | 0---100% |

上传代码后，将两节18650锂电池装入cobit的电池盒中，将电源开关拨到“ON”状态，打开串口监视器，波特率设置为9600，串口会将两节锂电池串联的电压值与电压百分比打印出来。

### 蓝牙接口示例



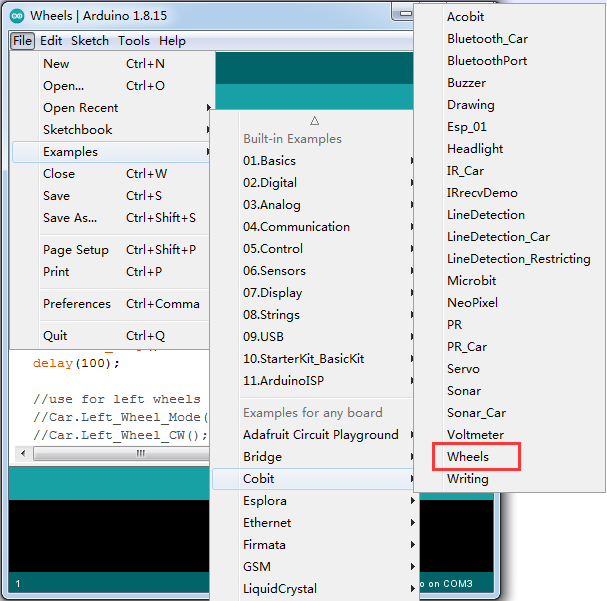
Cobit上预留有一个蓝牙接口，此接口为一个Soft serial port，波特率固定为115200，与蓝牙模块的连接方式为：

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Bluetooth port | V（5V） | G | D8 | D7 |
| 蓝牙模块 | VCC | GND | TX | RX |

上传代码后，将蓝牙与接口相连，然后使用手机蓝牙APP配对蓝牙，打开串口监视器，波特率设置为9600，按蓝牙APP上的任意按键，Arduino IDE的串口监视器将打印蓝牙APP的按键值。

注意：蓝牙必须处于串口透传模式才能传输数据给Cobit。

### 车轮转动示例

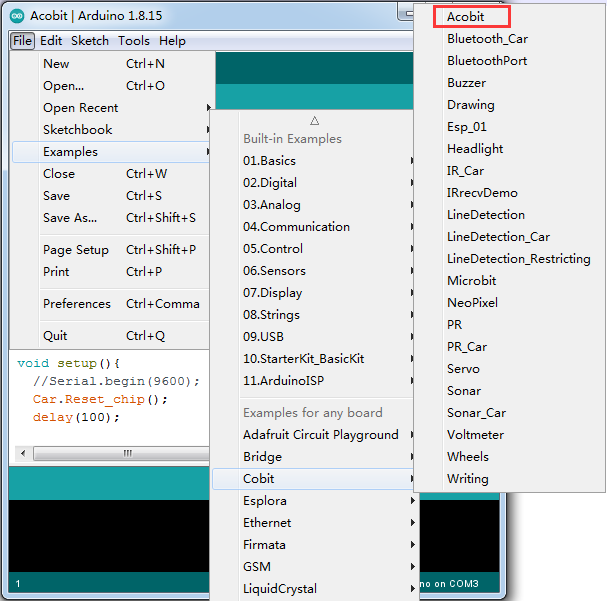


Cobit左右两边各有一个直径为65mm的轮子，使用两个步进电机作为轮子动力，可以控制轮子的转动角度与转动弧度，转动的角度精度达+/-0.9度，转动速度分为0---4档（0 = 0rpm，1 = 15rpm，2 = 30rpm，3 = 60rpm，4 = 120rpm）。

我们已将复杂的步进电机驱动代码编写成库文件形式，并预留有丰富的AIP接口，使用AIP接口可以非常简单、高效地实现各种应用，详细使用方法可以看”Wheels”示例代码”//”后面的解析。

上传代码后，Cobit左右两个电机先是顺时针转动5秒，再逆时针转动5秒，然后一直如此循环。

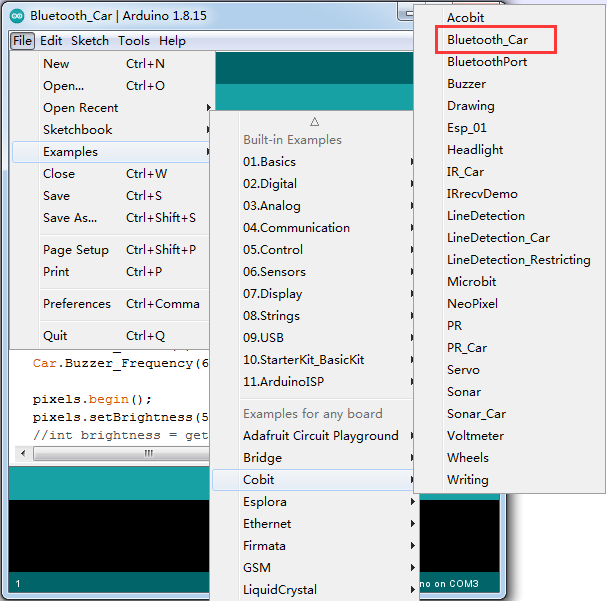
### Cobit的基本操作



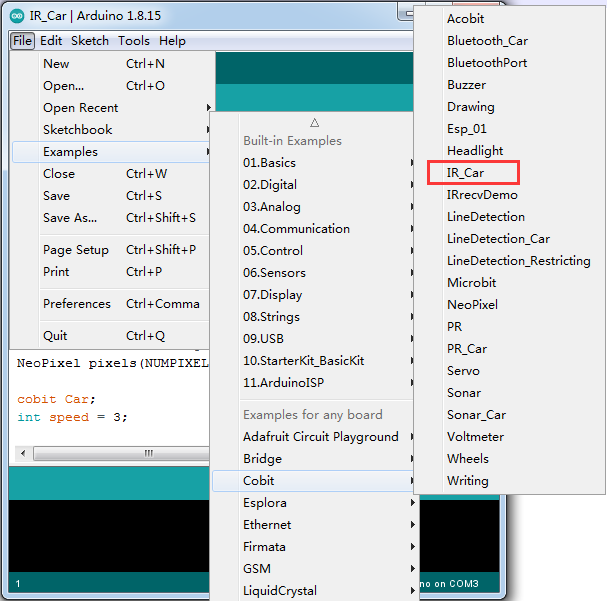
此示例包含cobit的基本行驶操作，如车的转向、行驶方向和它们对应使用的延时函数。

上传代码后，Cobit向做转360度，向右转360度，向前行驶200mm，再向后倒退200mm；之后向左连续转动2秒，向右连续转动2秒，向前行驶1秒，再倒退1秒，然后一直如此循环。

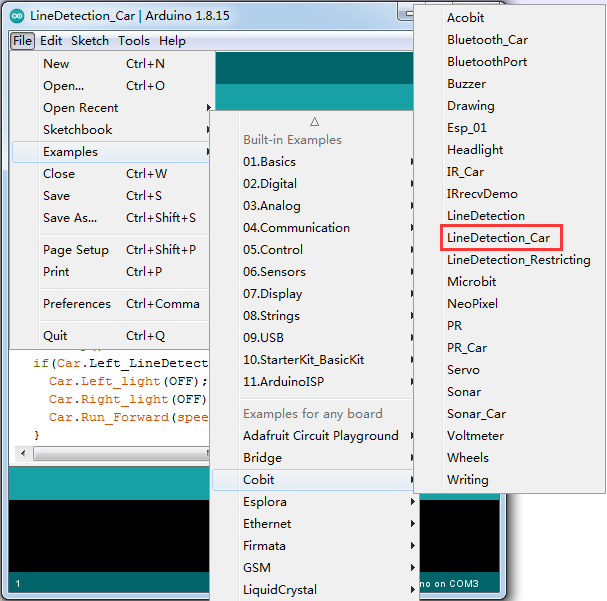
### 蓝牙遥控小车



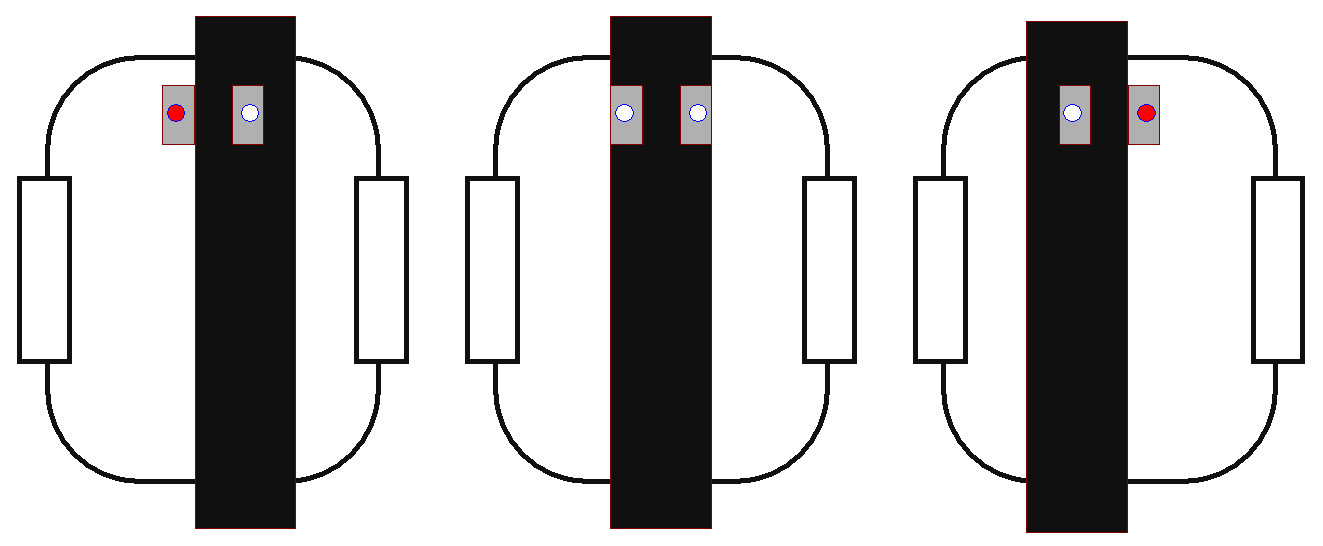
### 红外遥控小车



### 巡线小车

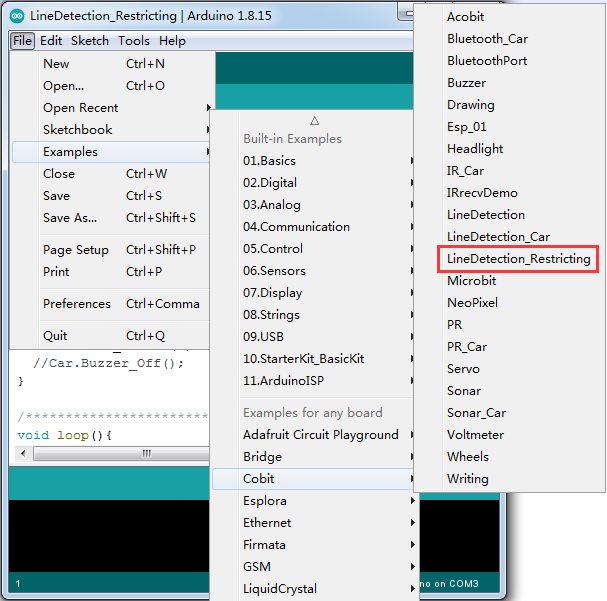


巡线小车使用的红外巡线传感器的功能，在白色的纸张上用2cm宽的黑色胶布贴成闭合的轨道形状，黑色物体对红外线有吸收效果，白色物体对红外线有反射效果，利用这个功能就可以让车一直沿着黑色线行驶。



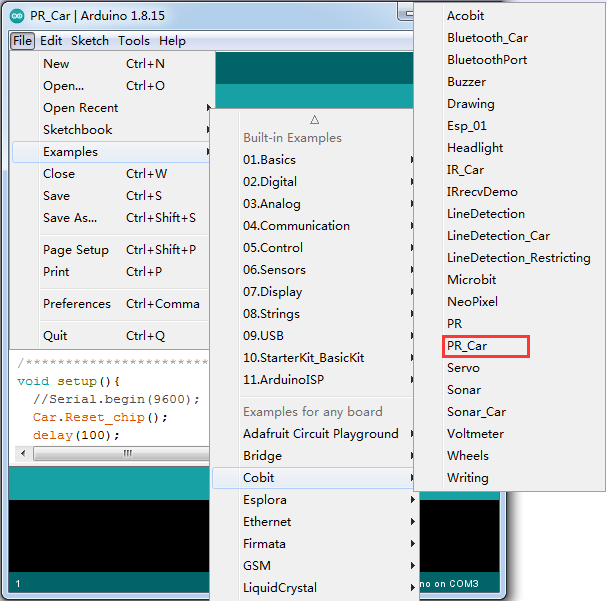
表示“0”，表示“1”。

### 防跌落小车

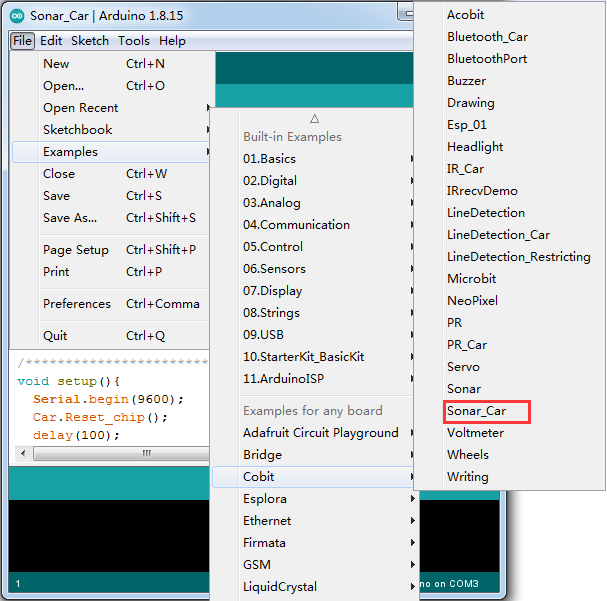


反跌落小车使用的红外巡线传感器的功能，将小车放在非黑色的桌面上，当小车在桌面上向前行驶时两个红外巡线传感器一直接收到反射信号，说明小车一直还在桌面内，当小车行驶到桌面边沿时，如果其中一个巡线传感器伸出了桌面，巡线传感器没形成反射信号，小车立即刹车停止，然后转向行驶。

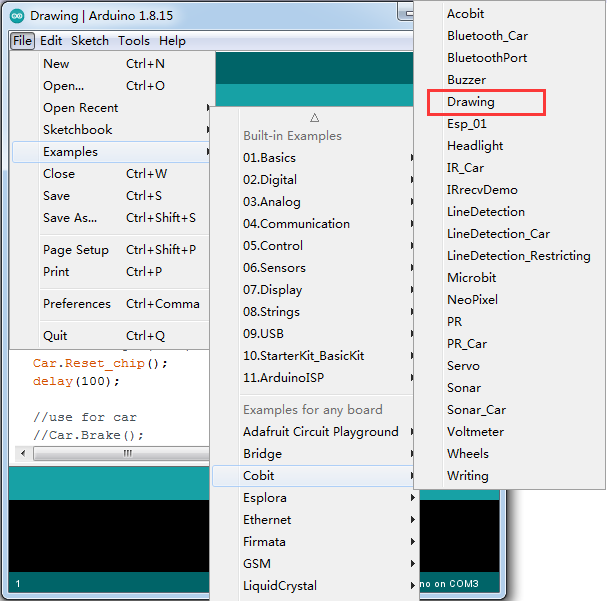
### 追光小车



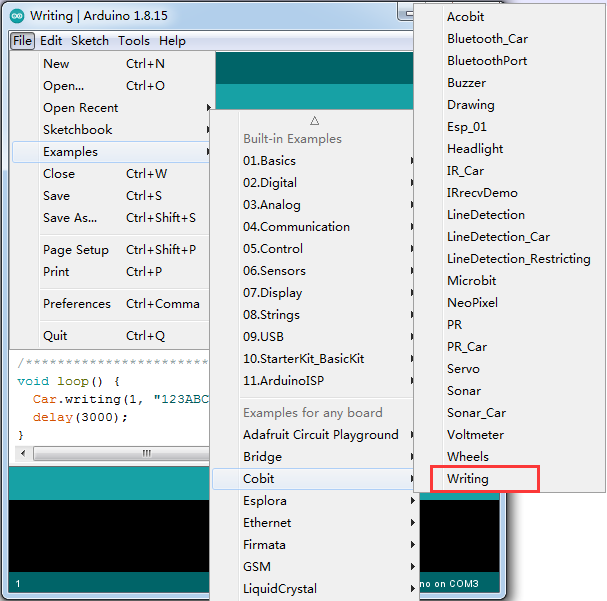
### Sonar避障小车



### 绘画小车



### 写字小车



# **Geting start with cobit for micro:bit**

Micro:bit教育基金会(The Micro:bit education Foundation)是2016年在英国成立的非营利组织，旨在激励每个孩子创造他们最好的数字未来。

主要宗旨是:

\*开发硬件和软件，激发年轻人对科技及其带来的机遇感到兴奋。

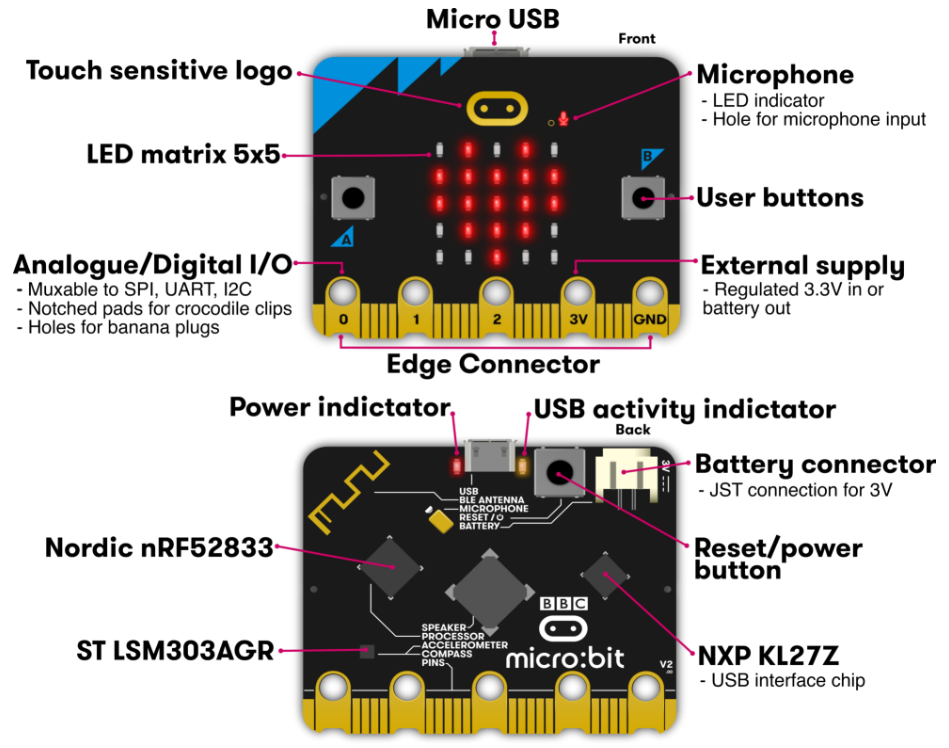
\*创建免费、用户友好的教育资源，支持教师提供有吸引力和创造性的课程。

\*与志同道合的伙伴合作，在全球范围内提供高影响力的教育项目。

Micro:bit官网：<https://microbit.org/>

## **Micro:bit 开发板**

Micro:bit有自己的专属开发板，分为V1与V2两个版本，现在主流使用的是V2版本，它功能强大，集成了大多数常用的传感器，保留有外扩的IO口，可以进行外扩展实验。



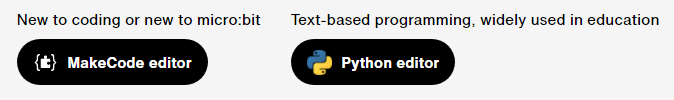
更多详细资料：<https://tech.microbit.org/>

## **Micro:bit 编辑器与编程语言**

**Editor selection：**

You can program your micro:bit in the online MakeCode block or Python text editors. Our **[Let's code](https://microbit.org/code/)** page helps you choose the one that's right for you.

Let’s code page: <https://microbit.org/code/>



**语法教程：**

Micro:bit平台提供了API与设备使用参考，包含了makecode、python、JavaScript的详细使用方法。

APIs: <https://makecode.microbit.org/reference>

Device: <https://makecode.microbit.org/device>

逻辑与数据类型参考。

[Blocks language](https://makecode.microbit.org/blocks): <https://makecode.microbit.org/blocks>

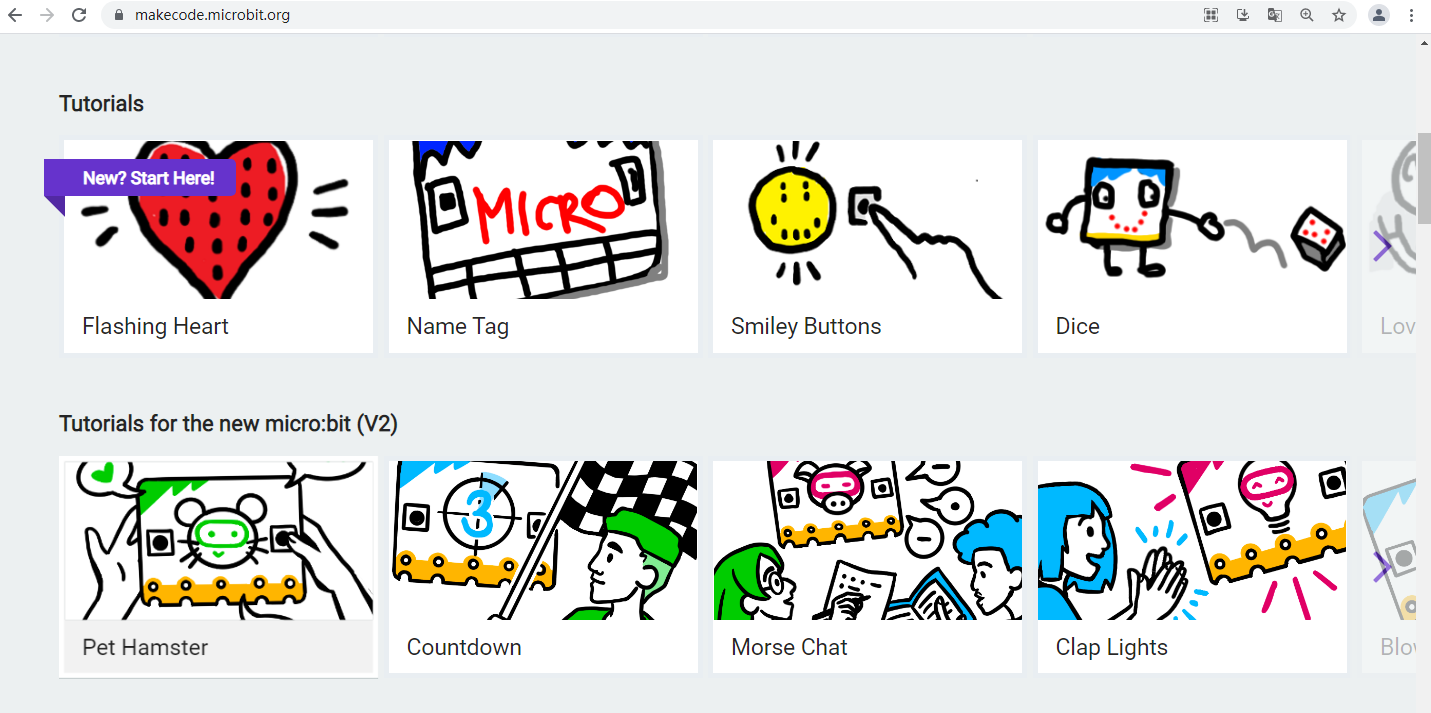
[JavaScript language](https://makecode.microbit.org/javascript): <https://makecode.microbit.org/javascript>

## **Get started micro:bit**

最好的学习方法就是动手，从模仿开始，跟着学习资料动手操作一遍。Micro:bit官网提供了丰富的视频教程，我们跟着视频动手操作一遍，熟悉Micro:bit开发的基本功能。

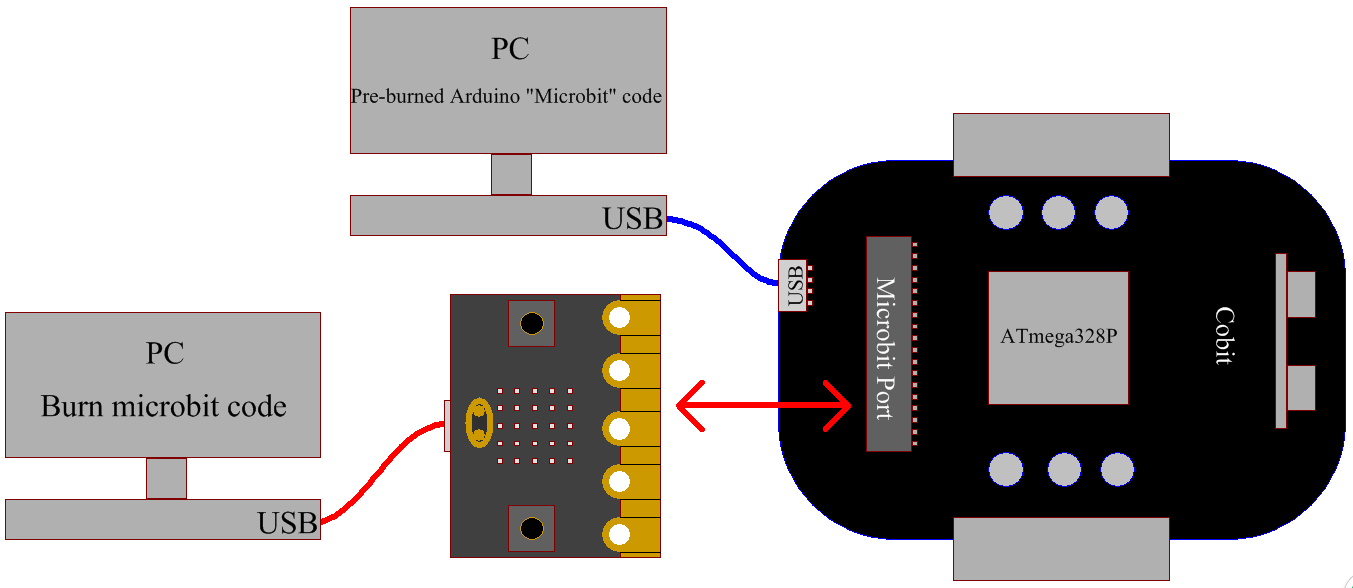
视频教程链接：<https://www.microbit.org/get-started/first-steps/introduction/>

更多视频教程也可以参考链接：<https://makecode.microbit.org/>



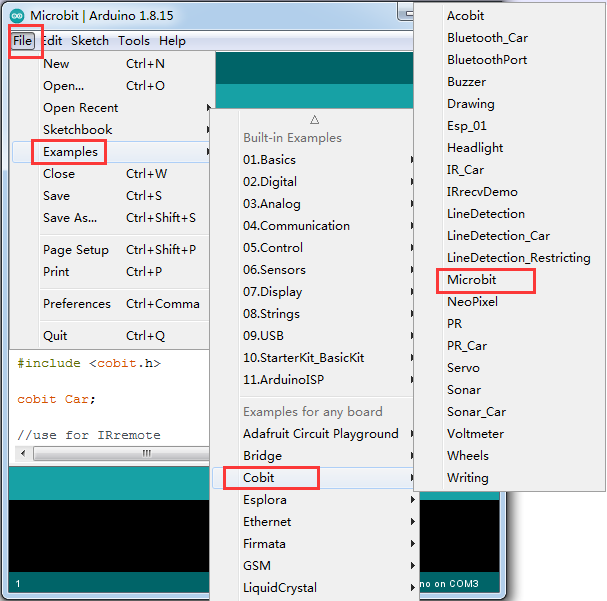
## **Cobit使用Micro:bit编程**

我们使用makecode为cobit开发了一个专属的扩展包，使得我们也可以使用micro:bit编程cobit，此micro:bit扩展包的使用是基于cobit烧录了arduino的”microbit”代码，否则micro:bit扩展包不起作用。

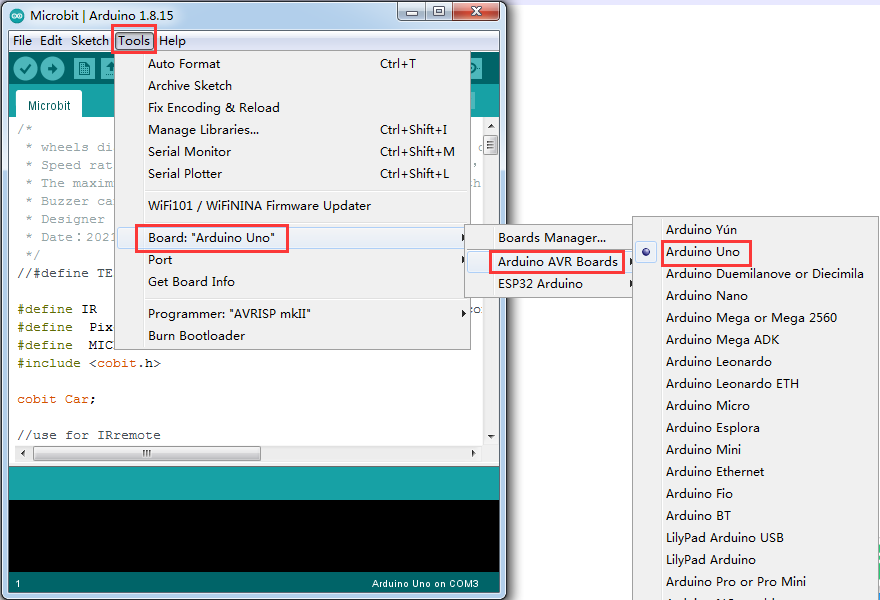


下载arduino ”microbit”代码，具体arduino IDE下载与Cobit库的安装请查看”Geting start with cobit for arduino”章节，”microbit”代码烧录步骤如下。

选择代码：

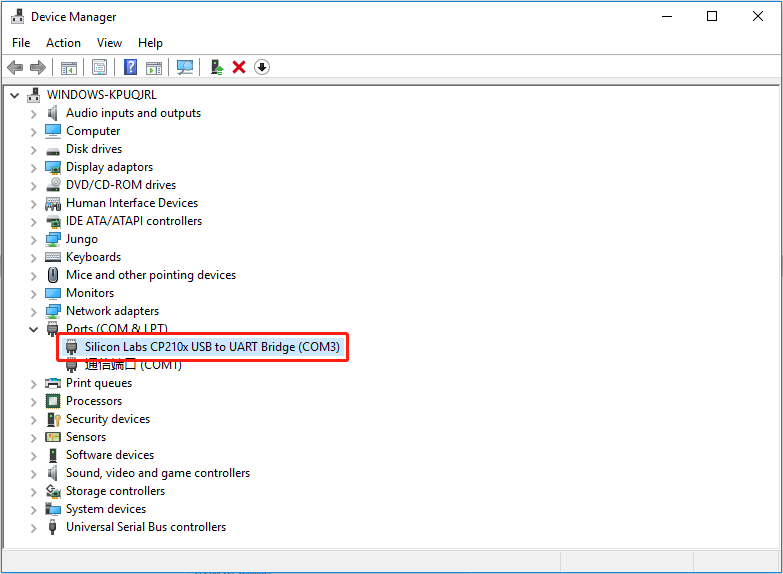


选择板型：

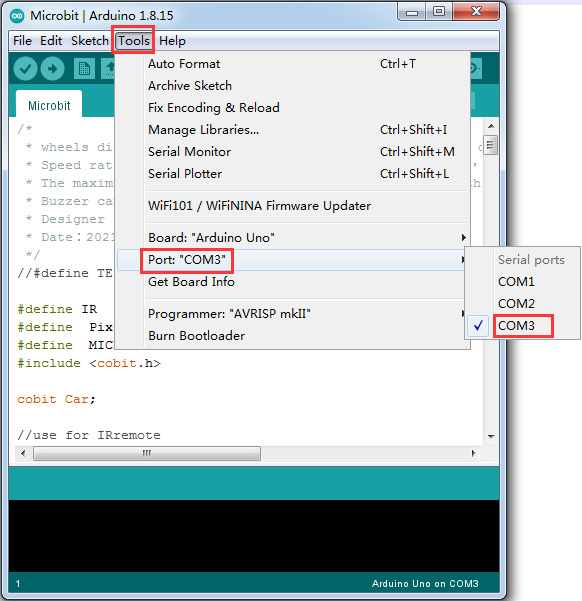


选择端口：

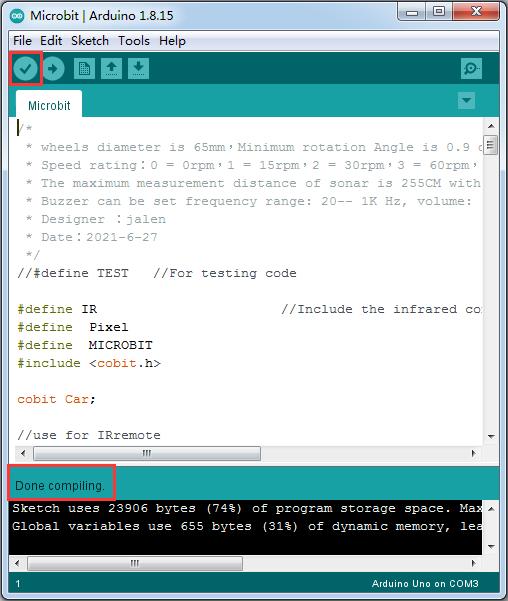
使用Type C USB线连接cobit与PC，打开PC的设备管理器（如果设备显示叹号，请参考”Install the USB-to-UART COM port driver”章节安装驱动）。



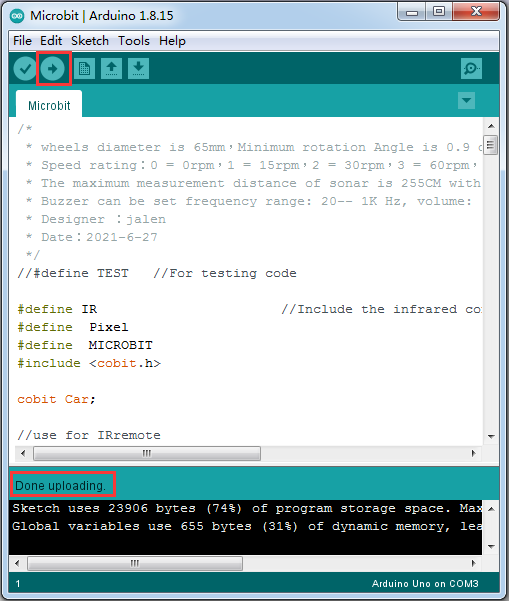
依据设备管理器的端口在arduino IDE中选择对应的端口。



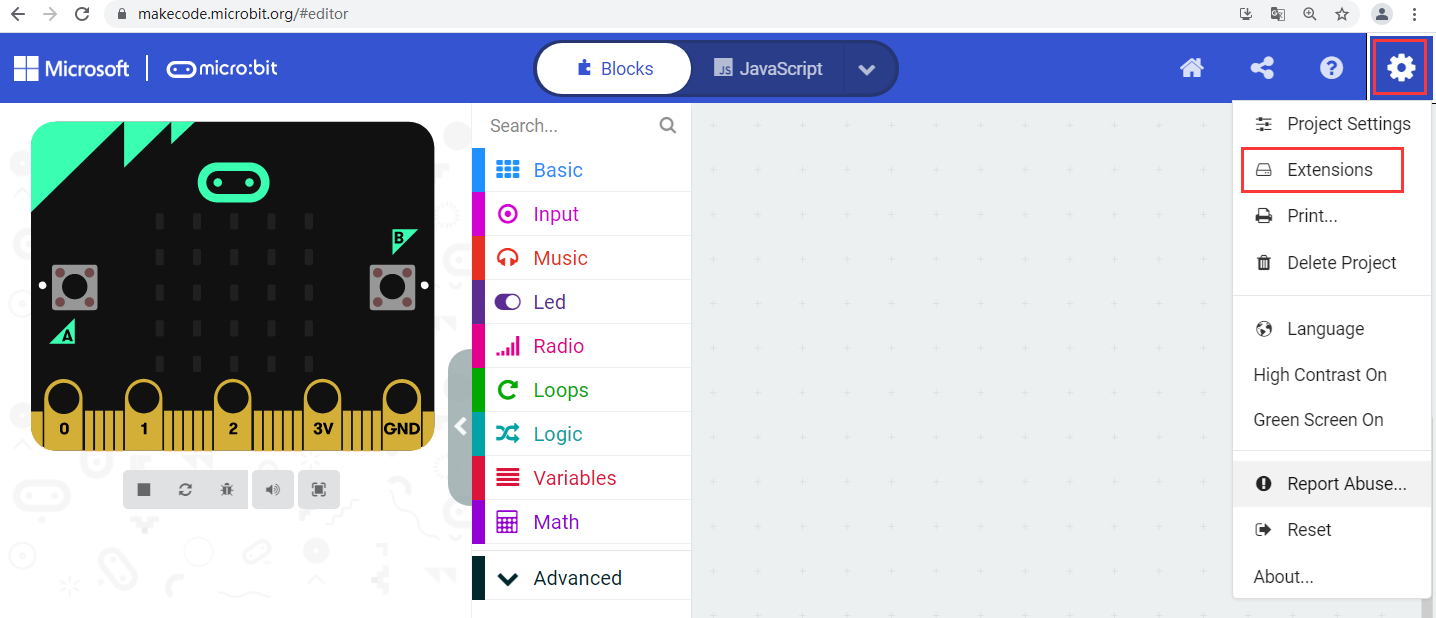
校验代码，如果代码有错误，此步骤无法通过，IDE底部的信息窗口打印橙色错误信息。

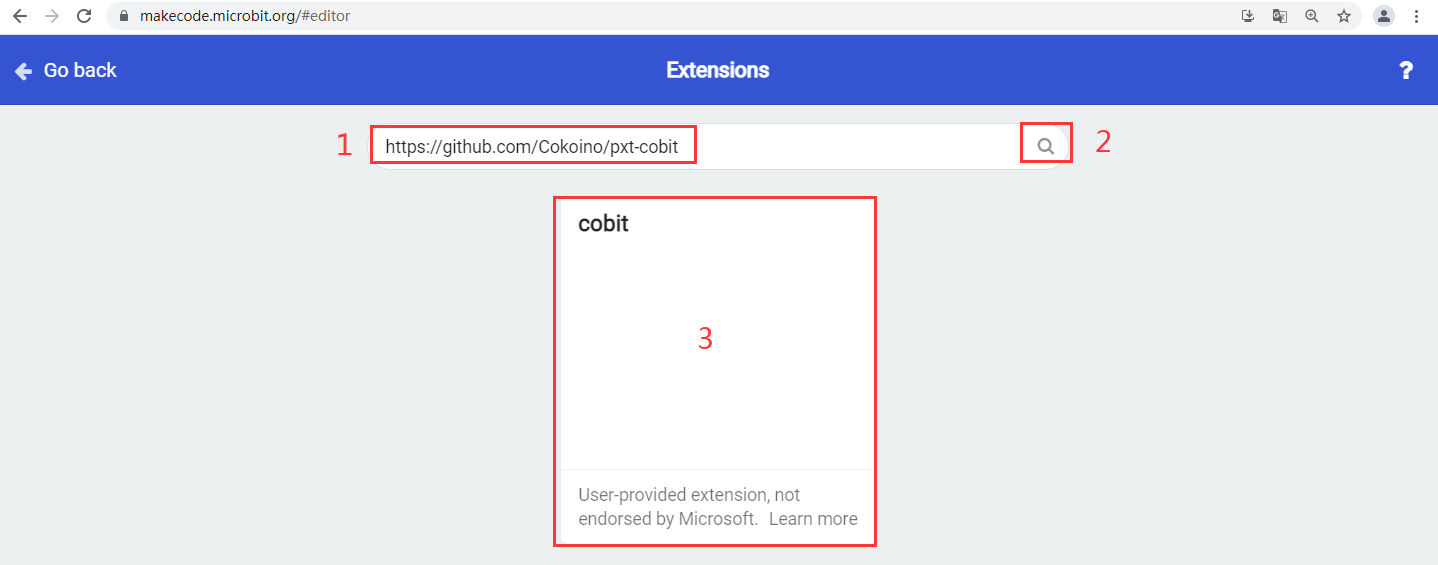


上传代码：

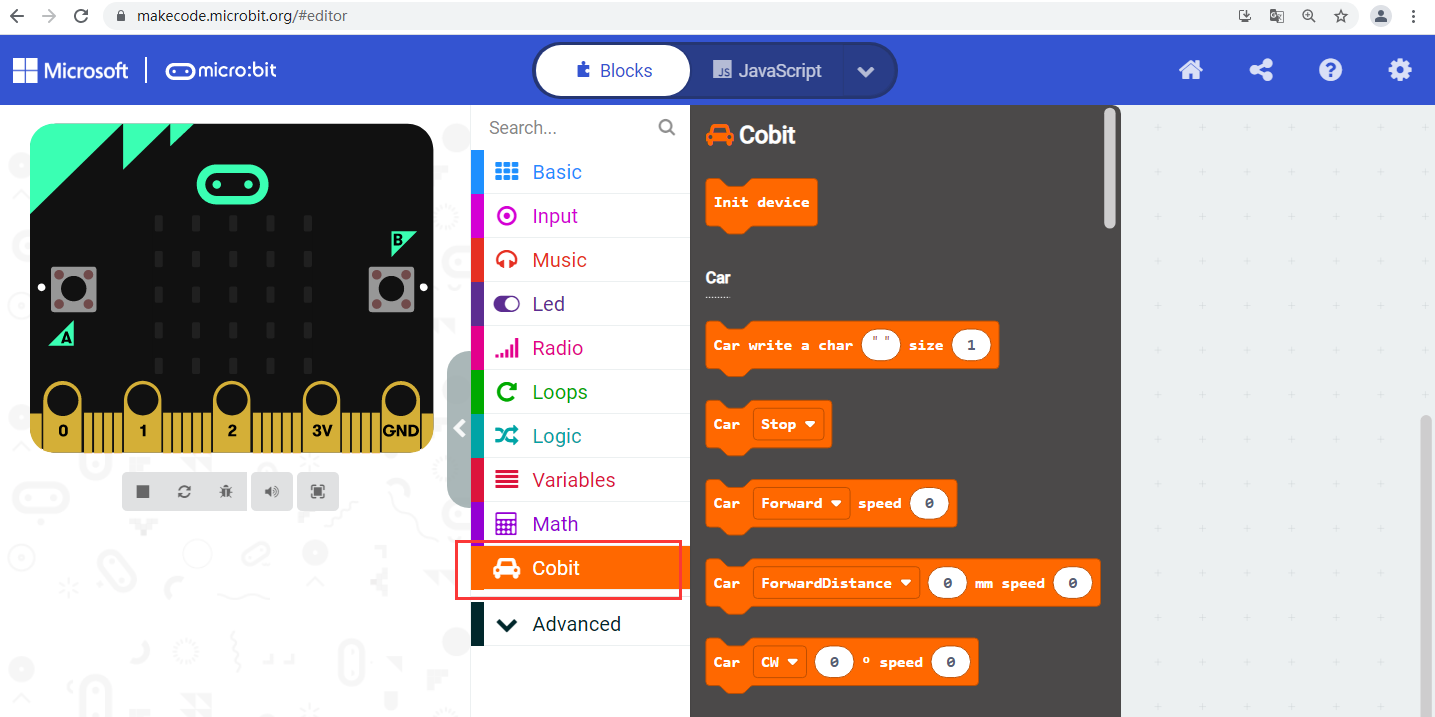


Micro:bit 的Cobit扩展包使用前需要进行加载，步骤如下：





基于micro:bit的cobit扩展包链接：<https://github.com/Cokoino/pxt-cobit>



至此，所有准备工作已好，可以使用makecode对Cobit进行编程。

## **基于Cobit的Makecode语句解析**

基于Cobit的Makecode语句全部集成在Cobit扩展包里，只需简单地拖拽到micro:bit代码编辑区域即可使用，语句解析如下:

|  |  |
| --- | --- |
| **语句** | **解析** |
|  | Cobit内部设备初始化语句，必须在“on start”中初始化一次。 |
| Car | |
|  | Cobit写字语句，第一个参数为ASCII可显示字符中的任一字符，第二个参数为字体的大小 |
|  | Cobit车停止运行语句。 |
|  | Cobit车连续行驶语句，第一个参数为方向，第二参数为速度。 |
|  | Cobit车精确距离行驶语句，第一个参数为方向，第二个参数为行驶距离，第三个参数速度。 |
|  | Cobit车精确转向语句，第一个参数问方向，第二个参数为转动角度，第三个参数为速度。 |
| Wheels | |
|  | Cobit车轮模式选择语句，第一个参数为轮子选择；第二个参数为模式选择，Step模式为步进模式，可以进行精确行驶，Continue模式为连续行驶模式，行驶距离与距离与转动角度不可控。 |
|  | Cobit车轮转向选择语句，第一个参数为轮子选择，第二个参数为转向选择。 |
|  | Cobit车轮速度选择，第一个参数为轮子选择，第二个参数为速递选择。 |
|  | Cobit车轮精确转动角度设置，第一个参数为轮子选择，第二个参数为转动角度设置。 |
|  | Cobit车轮精确转动距离设置，第一个参数为轮子选择，第二个参数为行驶距离设置。 |
|  | Cobit车轮状态选择，第一个参数为轮子选择，第二个参数为停止或者刹车状态选择。 |
| Sonar | |
|  | Cobit车的Sonar开关状态设置。 |
|  | 返回Cobit车的Sonar测量距离数据。 |
| IRremote | |
|  | 返回Cobit车的红外接收传感器接收到的数据。 |
|  | 红外遥控器的按键值。 |
| Buzzer | |
|  | Cobit车的蜂鸣器开关状态设置。 |
|  | 设置蜂鸣器的频率。 |
|  | 设置蜂鸣器的音量。 |
| Servo | |
|  | Cobit车舵机接口关状态设置。 |
|  | 设置舵机转动角度，第一个参数为选择接口，第二个参数为转动角度。 |
| Headlight | |
|  | 车头灯开关状态设置，第一个参数为车灯选择，第二个参数为开关状态选择。 |
|  | 车头灯亮度设置，第一个参数为车灯选择，第二个参数为亮度。 |
| PR | |
|  | 读取光敏电阻数据，参数为光敏电阻选择。 |
| LineDetection | |
|  | 读取巡线传感器的数据，参数为巡线传感器选择。 |
| RGB | |
|  | RGB灯亮度设置。 |
|  | RGB灯色卡，第一个参数为RGB灯选择，第二个参数为颜色选择。 |
|  | RGB灯任意颜色设置，第1个参数为RGB灯选择，第2---3参数为对应的红、绿、蓝颜色设置，通过调整这三参数的数值可以使RGB灯发出不同颜色的光。 |
| Voltmeter | |
|  | 读取Cobit车的电池电压。 |
|  | 读取Cobit车的电池百分比，0---100%映射为6.5V---8.4V。 |
| Port | |
|  | 设置Cobit车预留IO口的模式，第一个参数为IO口选择，第二个参数为IO模式选择。 |
|  | 读取IO输出电平设置，第一个参数为IO口选择，第二参数为IO输出电平设置。 |
|  | D11 IO口输出PWM信号，参数为0---255。 |
|  | 读取IO口的数字值，参数为IO口选择。 |
|  | 读取IO口的模拟值，参数为IO口选择。 |

## **实验课程**

**特别注意**，以下示例是基于cobit烧录了arduino的”microbit”代码，否则不起作用，具体arduino IDE下载与Cobit库的安装请查看”Geting start with cobit for arduino”章节，”microbit”代码烧录步骤如下。

### 车头灯示例

### 蜂鸣器示例

### 光敏电阻示例

### RGB灯示例

### 红外接收示例

### Sonar示例

### Servo示例

### 电压计示例

### 车轮转动示例

### 蓝牙遥控小车

### 红外遥控小车

### 巡线小车

### 防跌落小车

### 寻光小车

### Sonar避障小车

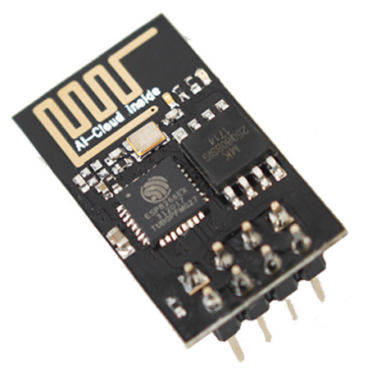
### 绘画小车

### 写字小车

# **Geting start with cobit for esp wifi**

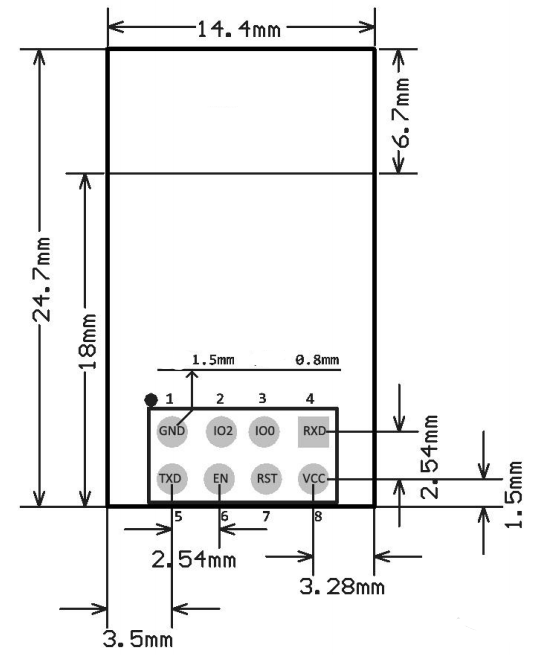
21世纪是物联网的时代，物联网给我们的生活带来了许多便捷，cobit上预留有给esp-01 wifi接口，使得cobit可以接入物联网。

## **ESP-01 wifi**



ESP-01可广泛应用于各种物联网场合，适用于家庭自动化、工业无线控制、婴儿监控器器、可穿戴电子产品、无线位置感知设备、无线定位系统信号以及其他物联网应用，是物联网应用的理想解决方案。

**接口定义：**



**Features**

- The smallest 802.11b/g/n Wi-Fi SOC module

- Low power 32-bit CPU, can also serve as the

application processor

- Up to 160MHz clock speed

- Supports UART/GPIO

- DIP-8 package for easy welding

- Integrated Wi-Fi MAC/BB/RF/PA/LNA

- Deep sleep current as low as 20uA

- Embedded lwIP protocol stack

- Supports STA/AP/STA + AP operation mode

- Support Smart Config/AirKiss technology

- UART baudrate up to 4Mbps

- General AT commands can be used quickly

- Supports remote firmware upgrade (FOTA)

## **Arduino与ESP -01 wifi的使用**

Arduino与ESP-01的使用是基于ESP-01烧录AT固件（出厂默认），ATmega328P通过software serial（D12=RX, D13=TX）发送AT指令控制ESP-01连接wifi发送与接收数据，从而实现物联网功能。

**Basic AT Commands**

[AT](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-at): Test AT startup.

[AT+RST](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-rst): Restart a module.

[AT+GMR](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-gmr): Check version information.

[AT+CMD](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-cmd): List all AT commands and types supported in current firmware.

[AT+GSLP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-gslp): Enter Deep-sleep mode.

[ATE](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-ate): Configure AT commands echoing.

[AT+RESTORE](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-restore): Restore factory default settings of the module.

[AT+UART\_CUR](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-uartc): Current UART configuration, not saved in flash.

[AT+UART\_DEF](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-uartd): Default UART configuration, saved in flash.

[AT+SLEEP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-sleep): Set the sleep mode.

[AT+SYSRAM](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-sysram): Query current remaining heap size and minimum heap size.

[AT+SYSMSG](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-sysmsg): Query/Set System Prompt Information.

[AT+USERRAM](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-userram): Operate user’s free RAM.

[AT+SYSFLASH](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-sysflash): Query/Set User Partitions in Flash.

[AT+RFPOWER](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-rfpower): Query/Set RF TX Power.

[AT+SYSROLLBACK](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-sysrollback): Roll back to the previous firmware.

[AT+SYSTIMESTAMP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-settime): Query/Set local time stamp.

[AT+SYSLOG](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-syslog): Enable or disable the AT error code prompt.

[AT+SLEEPWKCFG](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-wkcfg): Query/Set the light-sleep wakeup source and awake GPIO.

[AT+SYSSTORE](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-sysstore): Query/Set parameter store mode.

[AT+SYSREG](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Basic_AT_Commands.html" \l "cmd-sysreg): Read/write the register.

**Wi-Fi AT Commands**

[AT+CWMODE](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-mode): Set the Wi-Fi mode (Station/SoftAP/Station+SoftAP).

[AT+CWSTATE](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-state): Query the Wi-Fi state and Wi-Fi information.

[AT+CWJAP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-jap): Connect to an AP.

[AT+CWRECONNCFG](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-reconncfg): Query/Set the Wi-Fi reconnecting configuration.

[AT+CWLAPOPT](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-lapopt): Set the configuration for the command [AT+CWLAP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-lap).

[AT+CWLAP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-lap): List available APs.

[AT+CWQAP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-qap): Disconnect from an AP.

[AT+CWSAP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-sap): Query/Set the configuration of an ESP SoftAP.

[AT+CWLIF](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-lif): Obtain IP address of the station that connects to an ESP SoftAP.

[AT+CWQIF](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-qif): Disconnect stations from an ESP SoftAP.

[AT+CWDHCP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-dhcp): Enable/disable DHCP.

[AT+CWDHCPS](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-dhcps): Query/Set the IP addresses allocated by an ESP SoftAP DHCP server.

[AT+CWAUTOCONN](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-autoc): Connect to an AP automatically when powered on.

[AT+CWAPPROTO](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-approto): Query/Set the 802.11 b/g/n protocol standard of SoftAP mode.

[AT+CWSTAPROTO](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-staproto): Query/Set the 802.11 b/g/n protocol standard of station mode.

[AT+CIPSTAMAC](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-stamac): Query/Set the MAC address of an ESP station.

[AT+CIPAPMAC](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-apmac): Query/Set the MAC address of an ESP SoftAP.

[AT+CIPSTA](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-ipsta): Query/Set the IP address of an ESP station.

[AT+CIPAP](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-ipap): Query/Set the IP address of an ESP SoftAP.

[AT+CWSTARTSMART](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-starts): Start SmartConfig.

[AT+CWSTOPSMART](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-stops): Stop SmartConfig.

[AT+WPS](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-wps): Enable the WPS function.

[AT+MDNS](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-mdns): Configure the mDNS function.

[AT+CWHOSTNAME](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-hostname): Query/Set the host name of an ESP station.

[AT+CWCOUNTRY](https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/Wi-Fi_AT_Commands.html" \l "cmd-country): Query/Set the Wi-Fi Country Code.

更多信息请参考：

<https://docs.espressif.com/projects/esp-at/en/latest/AT_Command_Set/index.html>

## **实验课程**

**课程目的**

本实验要实现的是通过cobit的右侧光敏电阻获取当前环境的光强度数据，让后通过ESP-01将该数据上传到互联网的ThinkSpeak网站上，我们通过ThinkSpeak网站即可随时随地查看到该数据。

**工具**

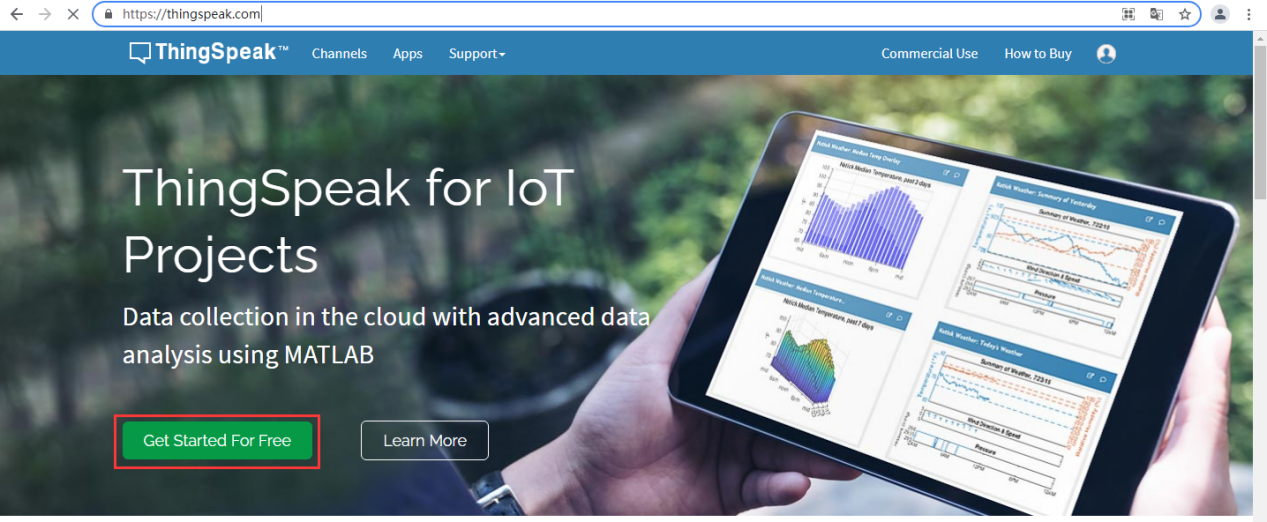
能上网的wifi网络、电脑。

**ThinkSpeak**

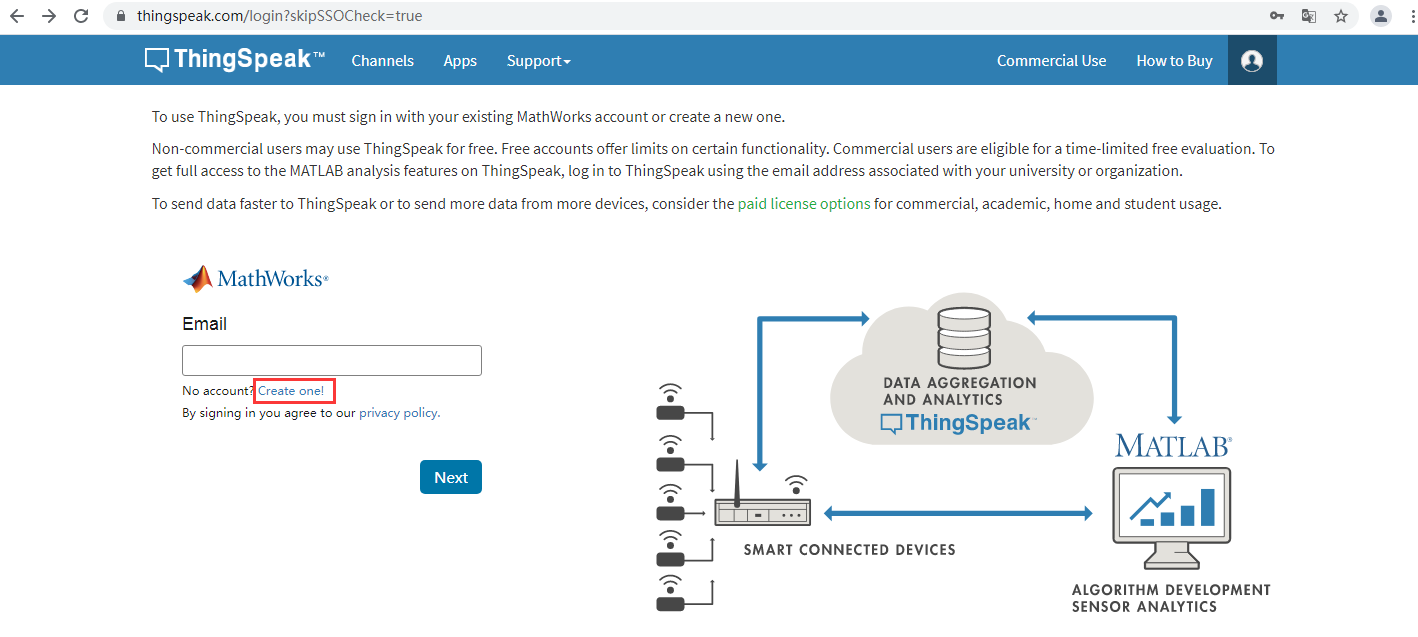
ThingSpeak is an IoT analytics platform service that allows you to aggregate, visualize and analyze live data streams in the cloud. ThingSpeak provides instant visualizations of data posted by your devices to ThingSpeak. With the ability to execute MATLAB® code in ThingSpeak you can perform online analysis and processing of the data as it comes in. ThingSpeak is often used for prototyping and proof of concept IoT systems that require analytics.

步骤1：注册ThinkSpeak账号

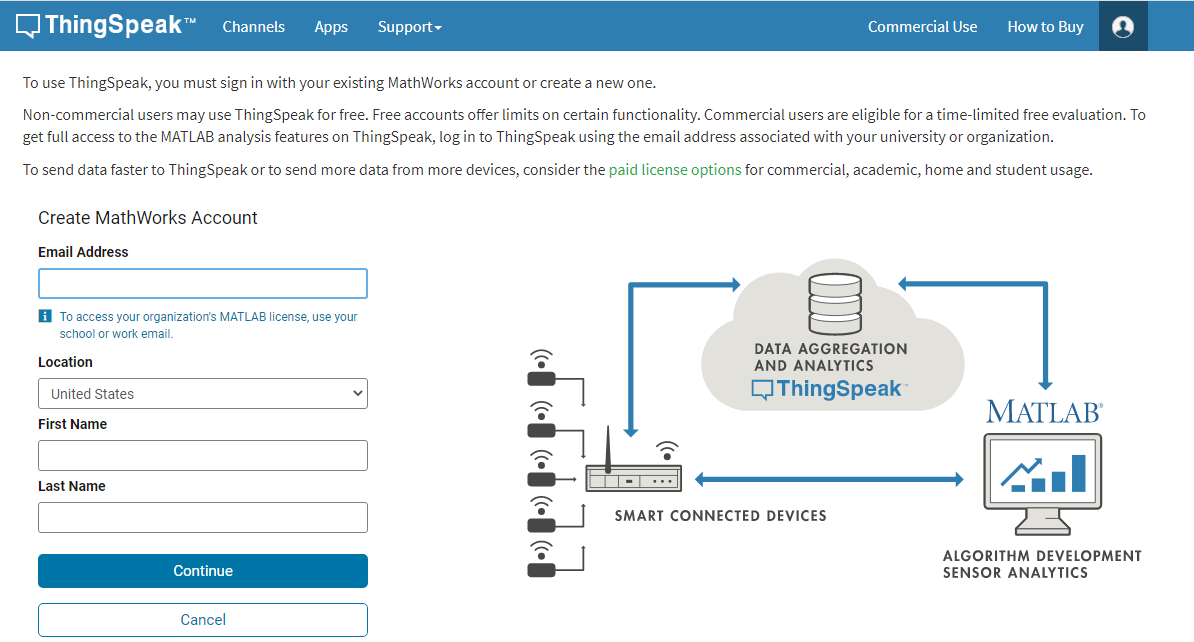
访问<https://thingspeak.com/>，然后点击“Get Started for Free”：

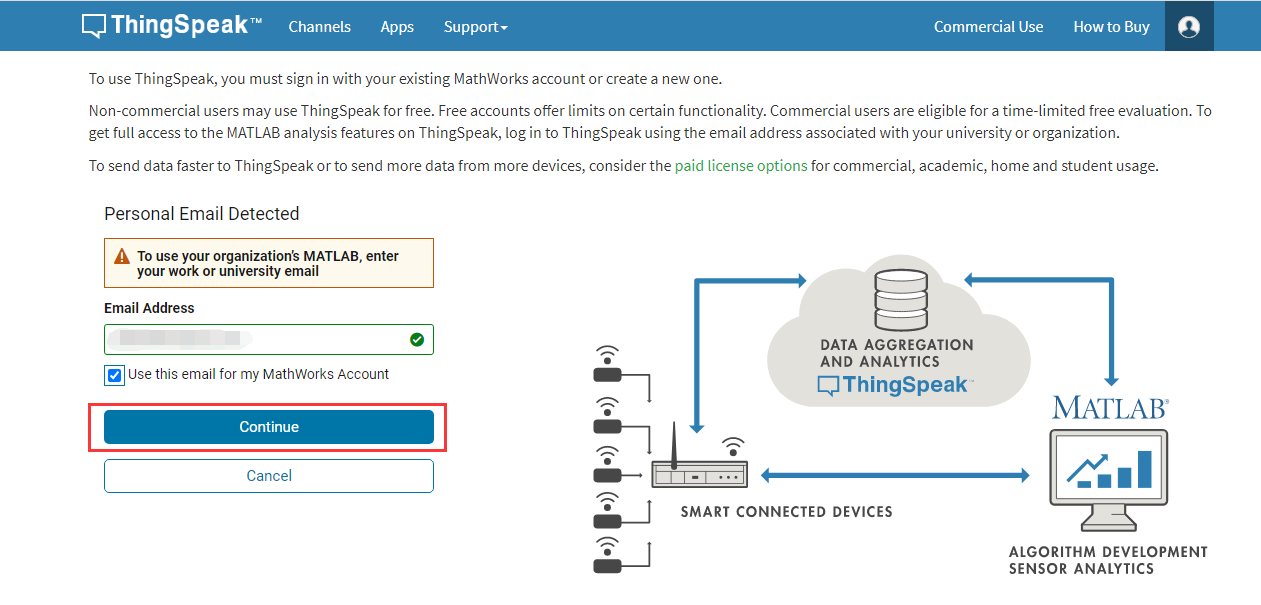


点击“Create one!”新建一个账号：

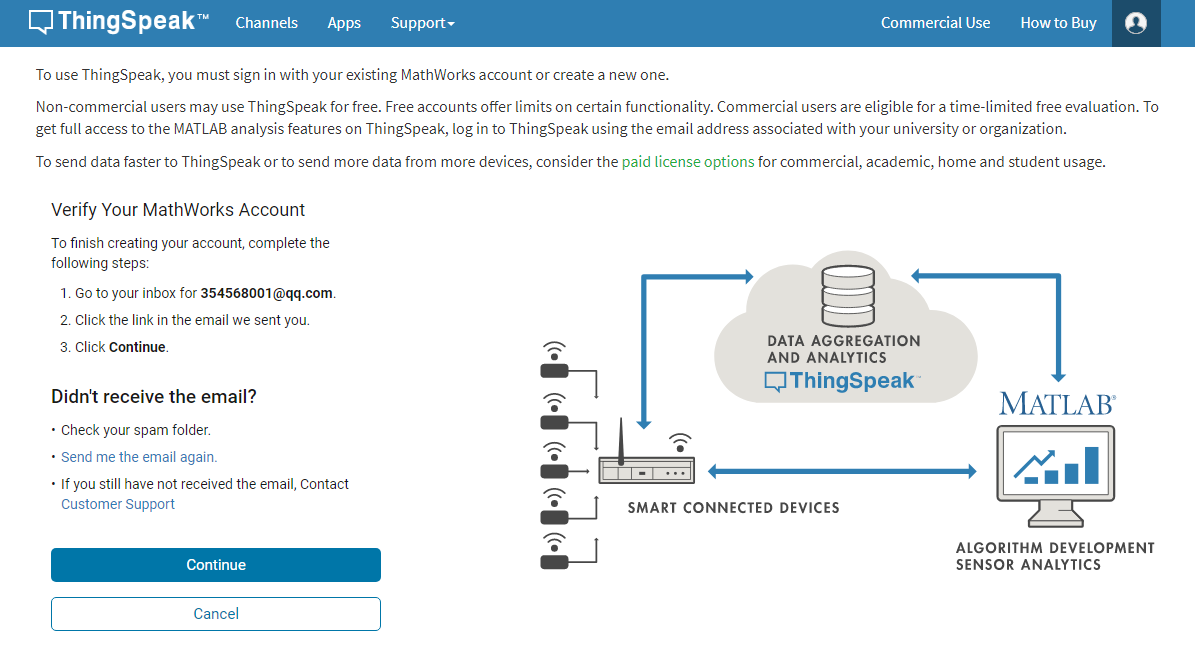


按要求填写相关信息，然后点击“Continue”，



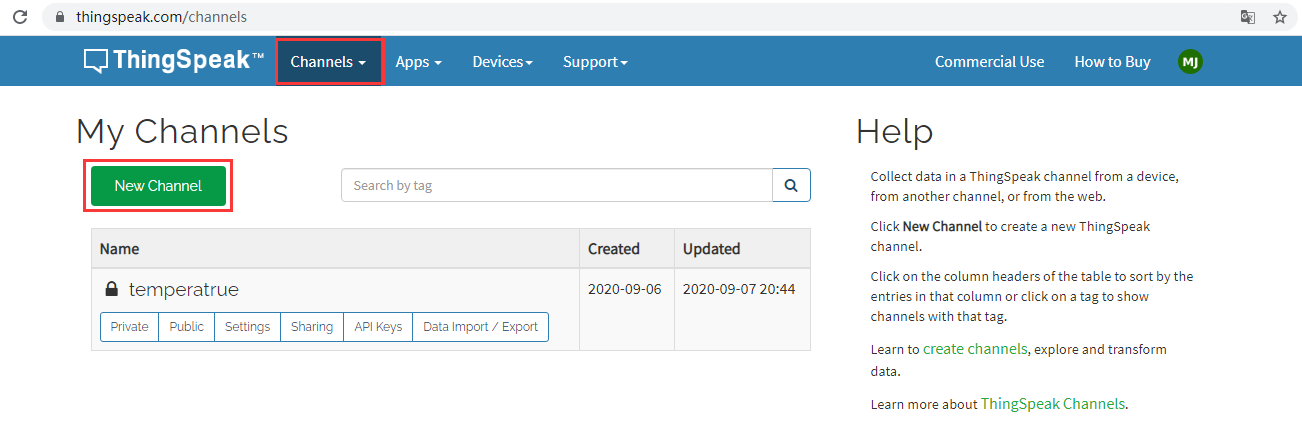


之后你的邮箱将会收到ThingSpeak发送的确认邮箱，按界面提示操作即可：

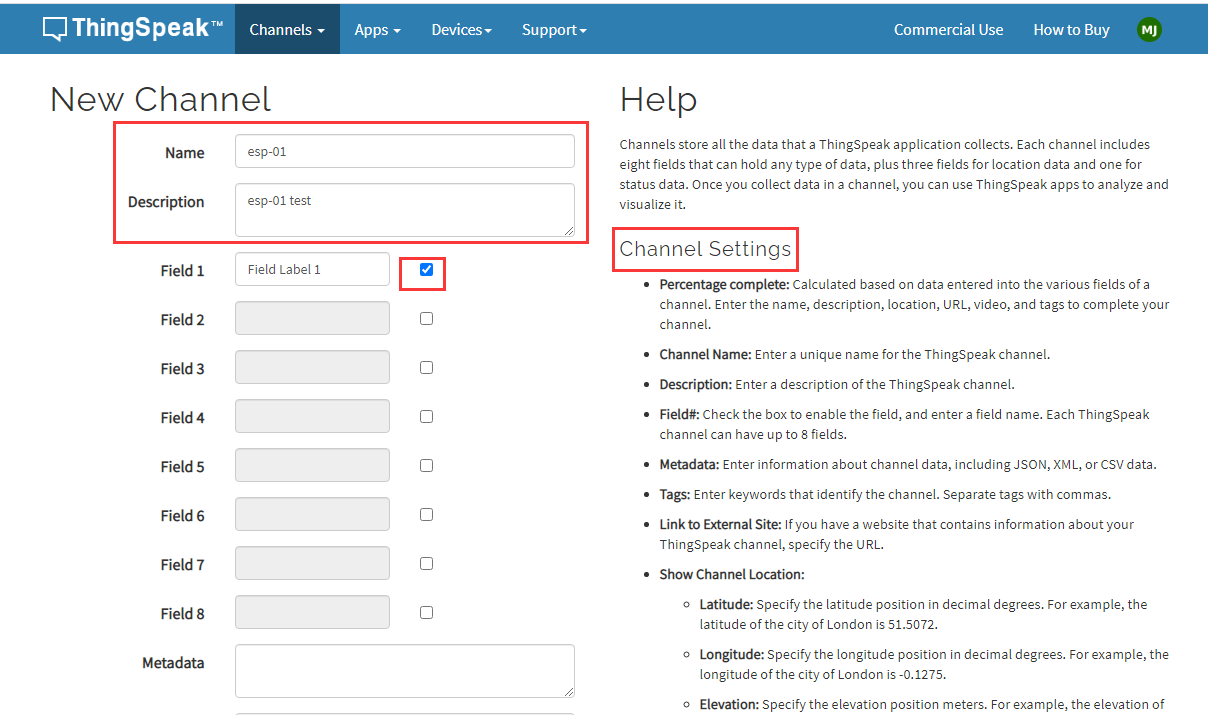


步骤2：创建ThingSpeak频道

账号新建成功后，点击“channels”-->“new channel”新建一个频道：



填写频道的名称与描述，因为我们只用到“field1”，所以这里只勾选“field1”，更多设置说明也可以参考页面右边的“Channel Settings”：

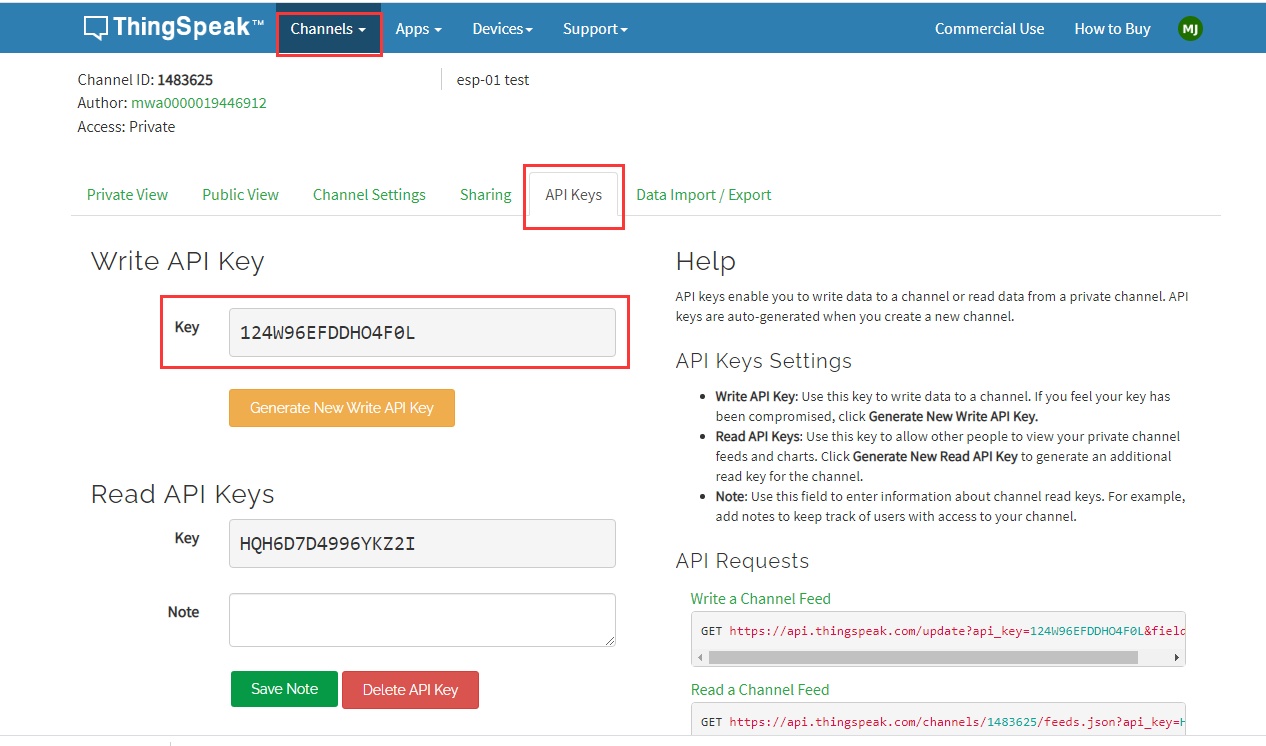


然后拖到页面底部，点击“Save Channel”即可完成创建频道：



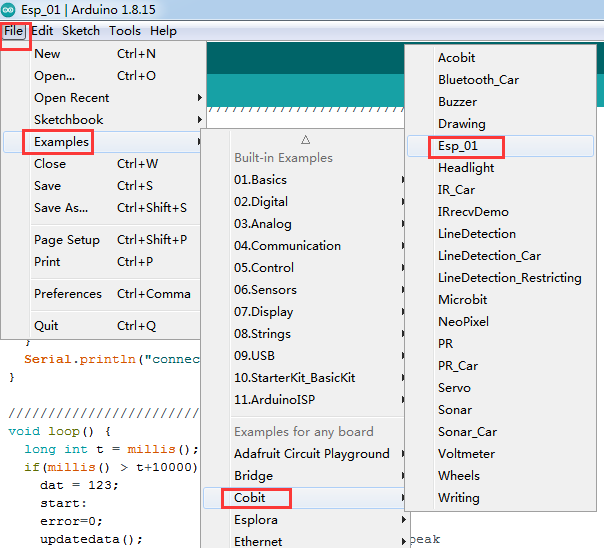
步骤3：获取ThingSpeak频道的API Keys

紧接上一步骤会自动跳转到“Channels”页面中上一步骤创建的频道，通过“Channels”下拉菜单可以选择“My Channels”可以选择你创建的其他频道（如果你创建了多个频道），然后选择“API Keys”，项目中我们只用到“Write API Key”，我们需要把他复制记录下来，后面代码使用到它：

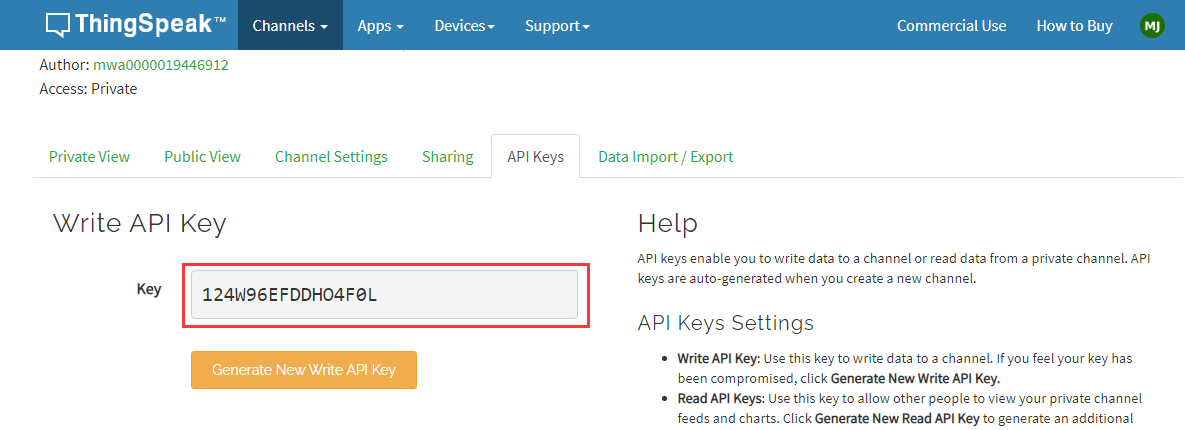


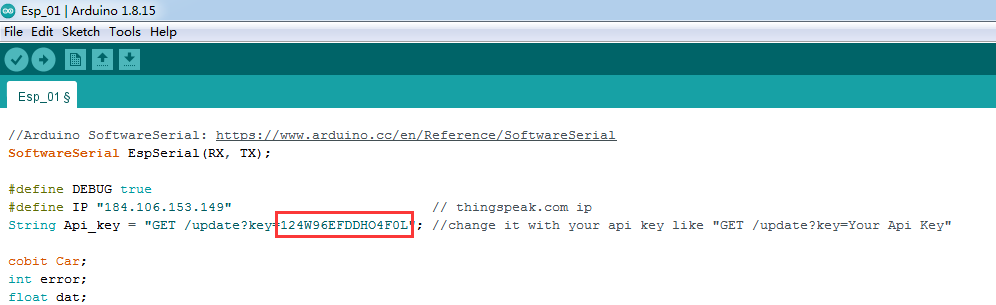
**Code**

Arduino IDE编辑代码可以轻松将数据通过ESP-01上传到ThingSpeak网站上，我们已经将代码包含在Cobit的Arduino库中，库的安装方法可以查看“Cobit使用arduino平台”章节。在确保arduino IDE已经安装了Cobit库，然后打开arduino IDE，选择“File”-->“Examples”-->“Cobit”-->“Esp-01”，即可打开示例代码：

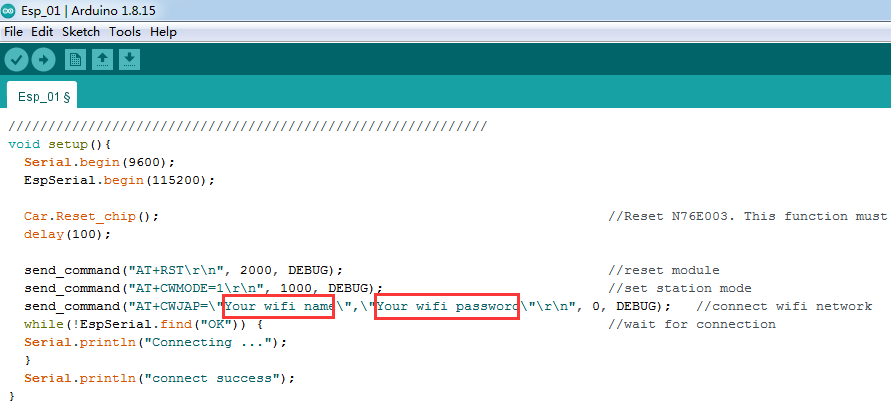


将示例代码的API Key修改为你账号上对应频道的AIP Key：



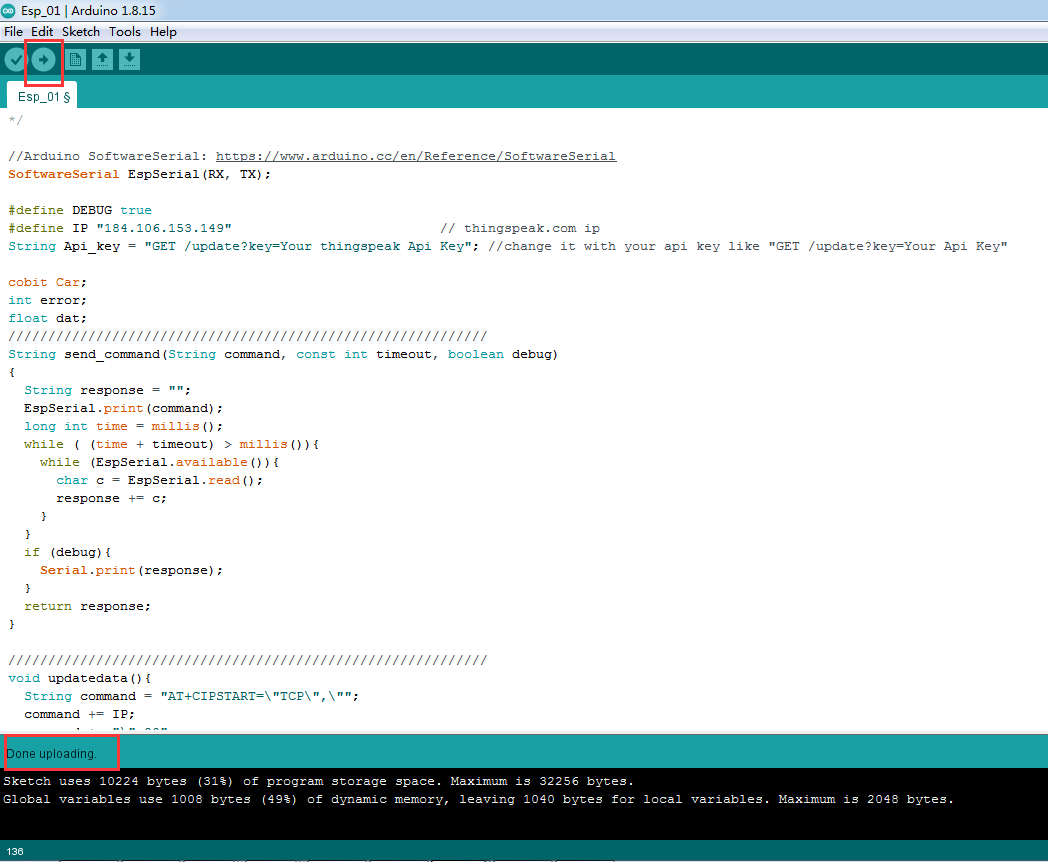


将示例代码的wifi名称与wifi密码修改为你自己的wifi名称与密码：

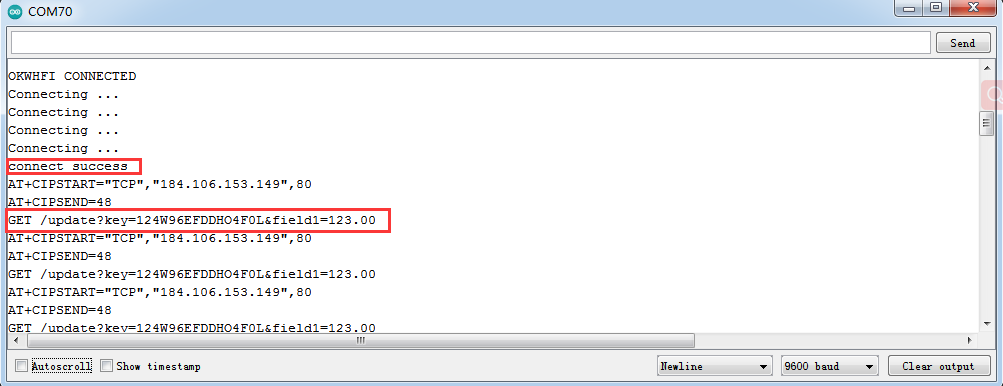


**上传代码**

Arduino传代码的详细方法可以查看“Cobit使用arduino平台”章节，选择好板型与端口号后，点击上传菜单键即可：

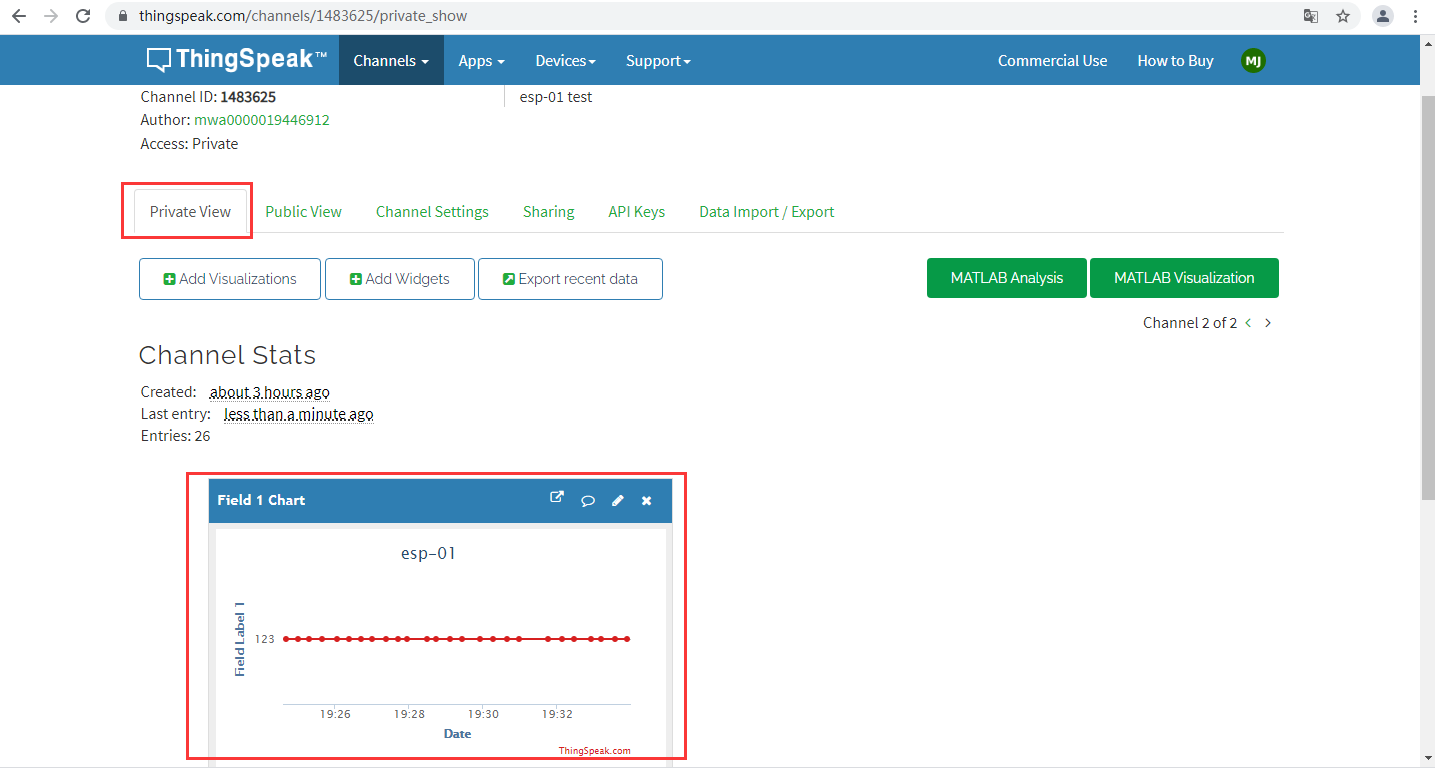


代码上传成功后，点击arduino IDE菜单，打开串口监视器，如果成功连接wifi串口会打印“connect success”，下图为成功将光敏数据“123.00”上传到了ThingSpeak网站。



提示：必须把关ESP-01插入cobit的对应接口上！

打开对应频道的“Pricate View”菜单，即可查看到上传的数据：



结束！

# **Trouble shooting**

## **Arduino**

**无法上传代码**

☛是否使用带有数据通信功能的USB数据线？

☛USB数据是否良好？

☛是否选择了正确的板型与端口？

**找不到端口号**

☛是否安装了CP2102驱动？

## **Micro:bit**

**无法上传代码**

☛目前Makecode仅支持编译小于或等于7KB的HEX代码，当大于7KB时会编译报错。

☛是否使用带有数据通信功能的USB数据线？

☛USB数据是否良好？

**盘符显示不对**

☛盘符显示为：MAINTENANCE (Normal is MICROBIT)

这是由于不小心按住micro:bit复位按键，然后给micro:bit上电，使micro:bit进入了更新固件模式。重新更新一下固件即可恢复正常，可以参考以下链接操作：

<https://microbit.org/get-started/user-guide/firmware/>

**不工作**

☛请检查micro:bit主板是否插反？

## **ESP-01**

**不工作**

☛Cobit代码是基于ESP-01的AT固件，如果重新录了固件会导致不工作，需要重新烧录出厂时的AT固件。

☛是否使用了非AT固件的ESP-01模块？

## **其他故障**

☛请检查组装是否正确？

☛请检查电池电量是否充足？

☛请检查使用的电池是否符合规格？

☛当电池电压过低时，会导致行走、写字错乱或者功能失常。

# **扩展与加入我们**

我们致力于开源硬件创客教育，诚心把我们的产品分享给大家，也希望能得到大家的认可。我们期盼的大家能多与我们沟通，找出我们产品的缺点，我们已经把我们所有的产品开源在github网站上，希望大家能经常互动，并加入到我们的团队中。

 <https://github.com/Cokoino>

✉[jex-spt@outlook.com](mailto:jex-spt@outlook.com)

 <http://cokoino.com>