阿里云 开发者社区 | 内容设计部 | AliOS Things

AliOS Things 快速开发指南

物联网开发从入门到精通

- 开发调试线上线下多样玩法
- 2大典型AliOS Things实践场景搭建
- 从零开始你的物联网开发之旅

藏经阁

阿里云开发者电子书系列

阿里云开发者“藏经阁”

海量免费电子书下载

加入钉群

获取更多前沿资讯

加入钉群

与更多人进行技术交流

目录

初识 AliOS Things

AliOS Things 快速入门帮助您初识 AliOS Things 系统，并实现从零开始的一个入门操作。

一、什么是 AliOS Things

AliOS Things 发布于 2017 年杭州云栖大会，是 AliOS 家族旗下，面向 IoT 领域的高可伸缩物联网操作系统。

AliOS Things 致力于搭建云端一体化 IoT 基础设施，具备极致性能、极简开发、云端一体、丰富组件、安全防护等关键能力。AliOS Things 支持多种多样的设备连接到阿里云 IoT 平台，可广泛应用在智能家居、智慧城市、工业，新出行等领域。

关键特性

o 即插即用的连接和丰富的服务。

o 支持 uMesh 即插即用网络技术，设备上电自动连网。

o 差分+安全 OTA 升级。

o 差分+安全 OTA 升级。

o 差分增量包升级。

o 安全数字签名。

o 安全下载通道。

o 断点续传。

o 乒乓升级。

o 版本回溯。

o 全面彻底的安全保护。

o 提供系统和芯片级别安全保护。

o 支持可信运行环境（支持 ARMV8-M Trust Zone）。

o 支持预置 ID2 根身份证和非对称密钥以及基于 ID2 的可信连接和服务。

o 高度优化的性能。

o 内核支持 Idle Task，内存资源消耗低，RAM 小于 1 KB，ROM 小于 2 KB，提供硬实时能力。

o 提供 Yloop 事件框架以及基于此整合的核心组件，避免栈空间消耗，核心架构良好支持极小 footprint 的设备。

o 极简开发

o 基于 Linux 之上的轻量虚拟化环境，提供在 Linux 平台上开发与硬件无关的 IoT 应用和软件库，使用 GDB、Valgrind、SystemTap 等 PC 平台工具诊断开发问题。

o 提供 IDE，支持系统、内核行为 Trace， Mesh 组网图形化显示。

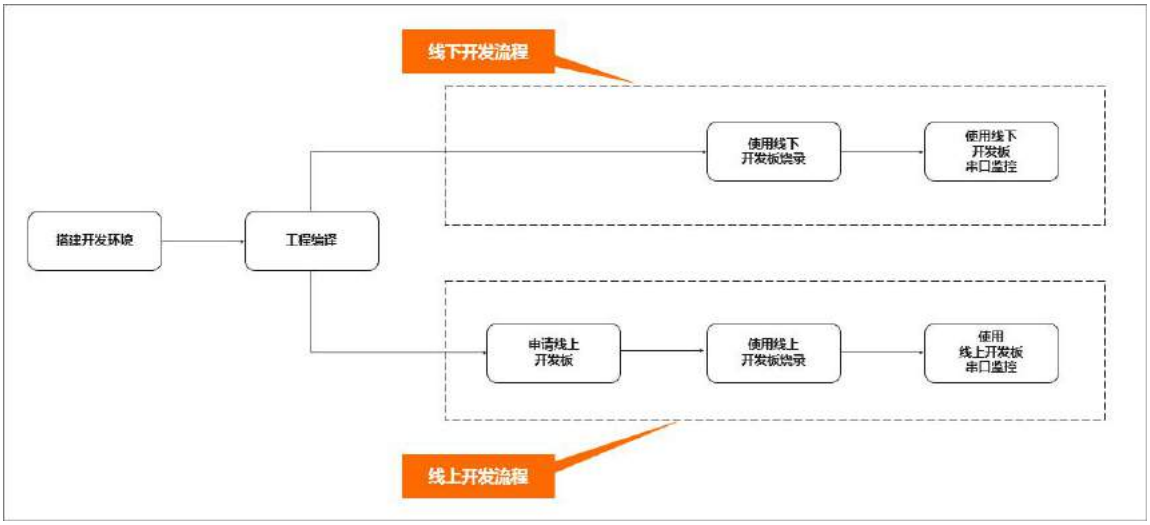
o 提供 Shell 交互，支持内存踩踏、泄露、最大栈深度等各类侦测。

o 提供面向组件的编译系统以及 aos-cube 工具，支持灵活组合 IoT 产品软件栈。

o 提供包括存储（掉电保护、负载均衡）在内的各类产品级别的组件。

二、两大开发流程

AliOS Things 编译完成后，可以使用线上或者线下开发板烧录。调试完成后，即可应用到您的实际业务中。 开发板开发流程图如下：



开发板类型	流程说明	适用场景
线下开发板	1、安装开发环境 2、项目编译 3、固件烧录 4、调试	真实开发环境。
线上开发板	开发的流程与上述使用线下开发板的流程基本一致。 主要的区别在于固件烧录前，需要先申请线上开发板，操作步骤请参见使用线上开发板进行开发调试。	线上体验。线上开发流程主要适合当您手上没有现成可用的实体开发板时，可以使用线上的开发板来调试验证您的程序。 说明 在实际开发中，您仍然需要使用线下开发板进行开发。

开发前准备

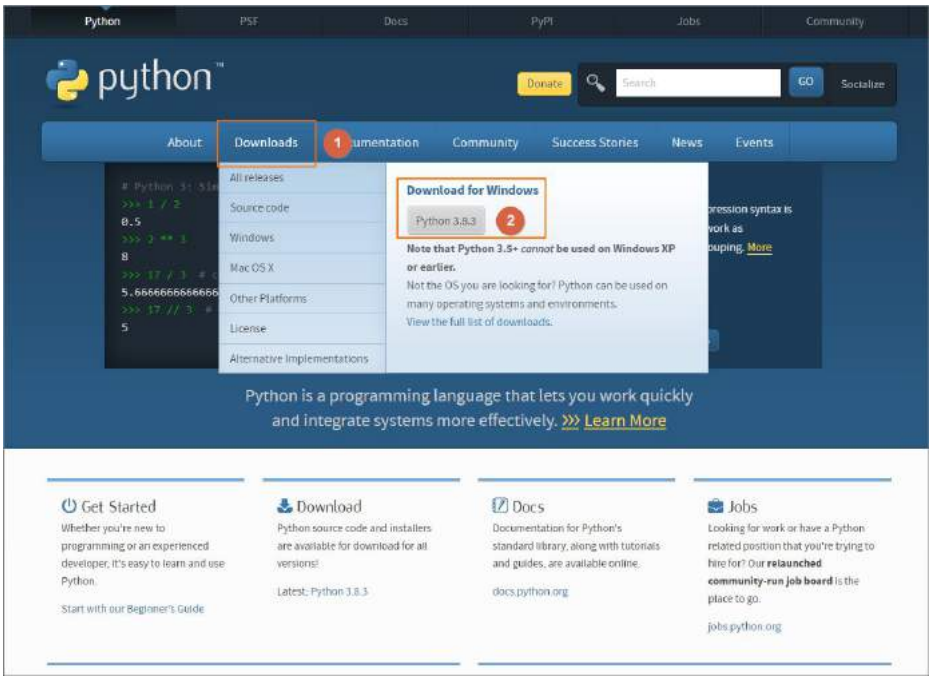
在运行 AliOS Things 系统之前，您需要做好一系列准备工作，包括搭建环境、安装驱动设备、下载 AliOS Things 系统源码、安装开发工具 AliOS Studio 等。本文详细介绍如何完成这些准备工作。

一、背景信息

AliOS Things 系统依赖 Python、pip、Git 环境。AliOS Studio 是 AliOS Things 的图形化开发组件， 是一套基于 VSCode（Visual Studio Code）的开发工具，支持 Windows，Linux，macOS。AliOS Things 系统源码目前最新版本是 3.1.0，支持裁剪定制，根据需求选择需要的组件。

步骤一：安装 Python

1. 下载 Python 安装程序。
   a. 使用浏览器打开 Python 官网。
   b. 单击导航栏的 **Downloads**，然后单击 **Python 3.8.3** 开始下载，如下图所示。

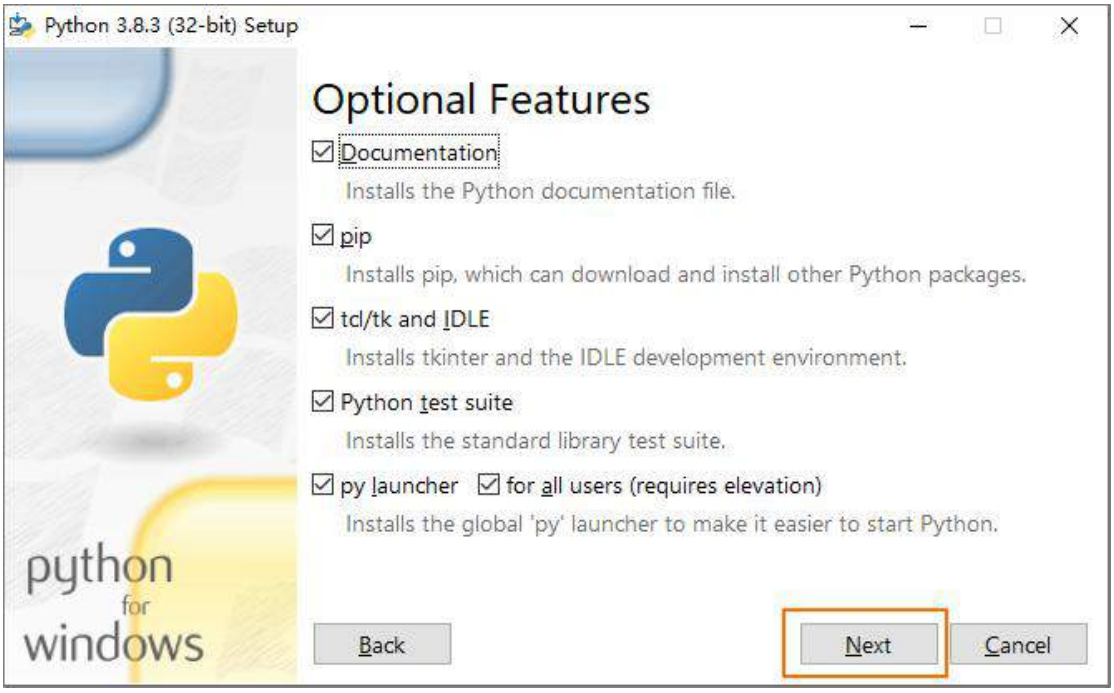


2. 安装 Python。

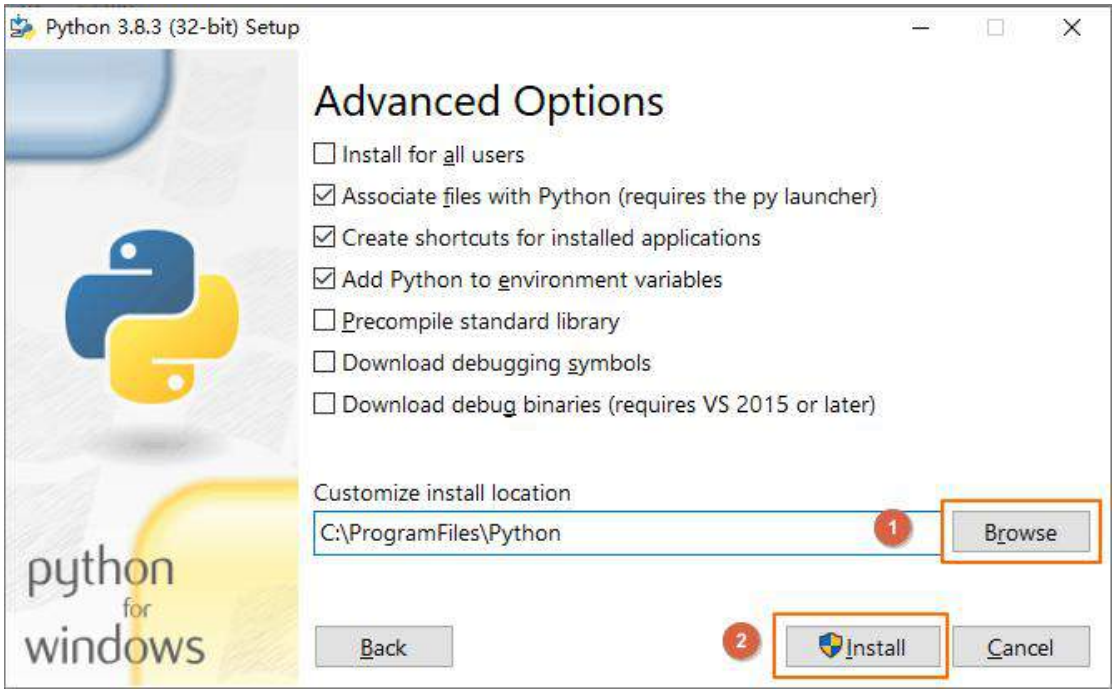
a. 双击 **Python 3.8.3.exe** 开始安装 Python。

b. 勾选 **Add Python 3.8 to PATH**，然后单击 **Customize installation** 进行自定义安装。

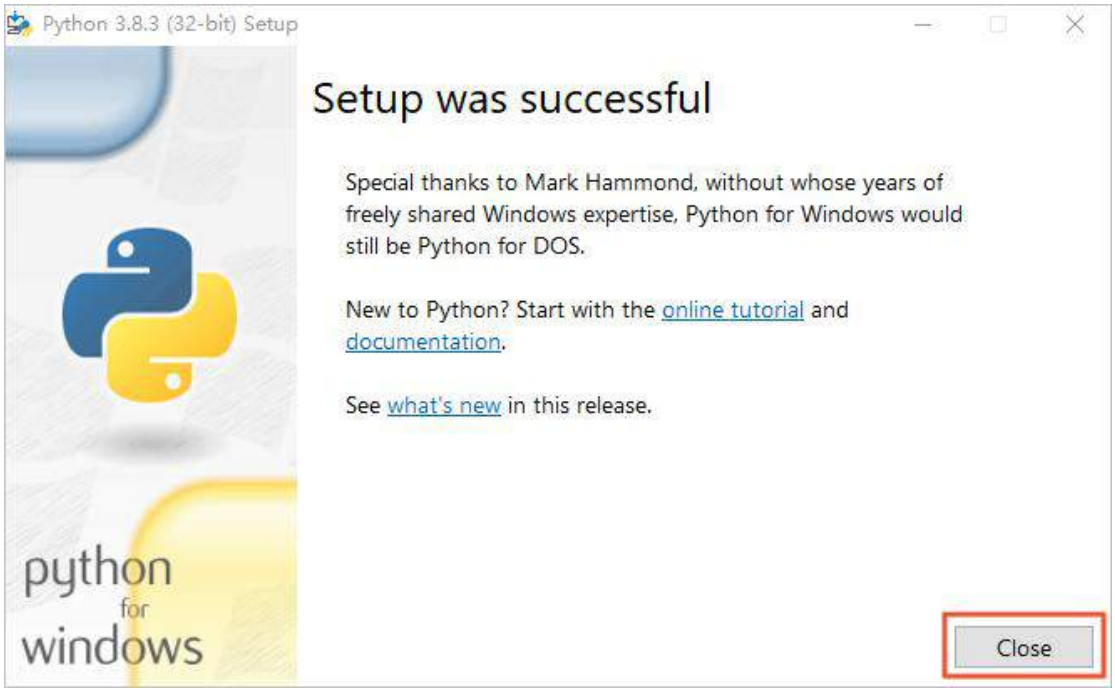
c. 使用默认配置，单击 **Next**。



d. 单击 **Browse** 选择自定义安装目录。然后单击 **Install** 开始安装。

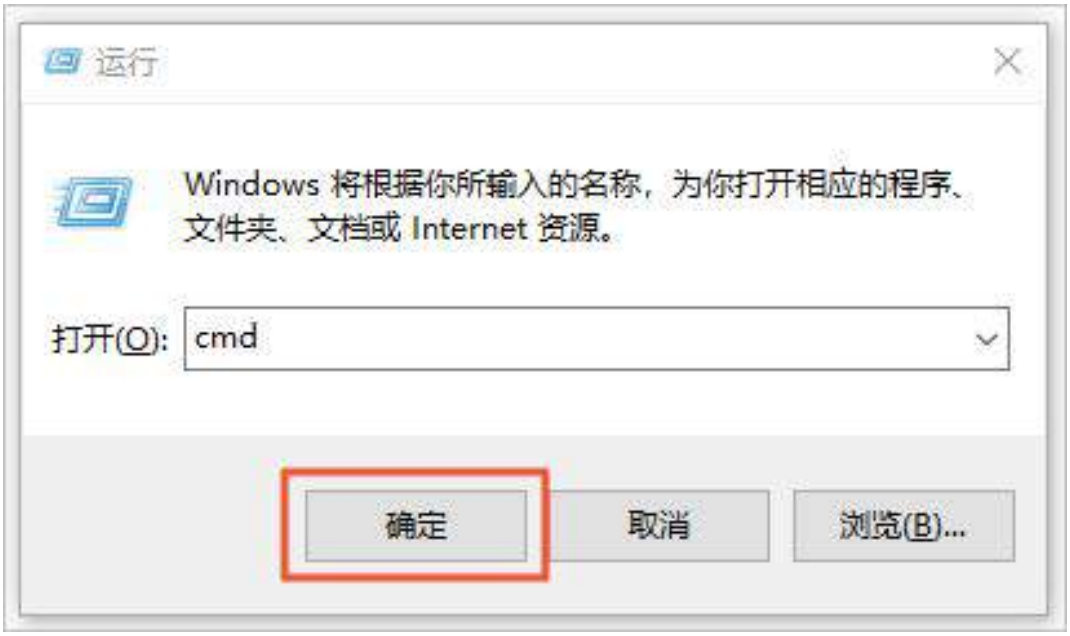


e. 单击 **Close**， **Python** 安装完成。



3. 验证 Python 是否安装成功。

a. 使用 **Win+R** 快捷键打开**运行**窗口，在输入框里输入 **cmd**。然后单击**确定**进入命令窗口。



b. 执行以下命令，查看 Python 是否安装成功。

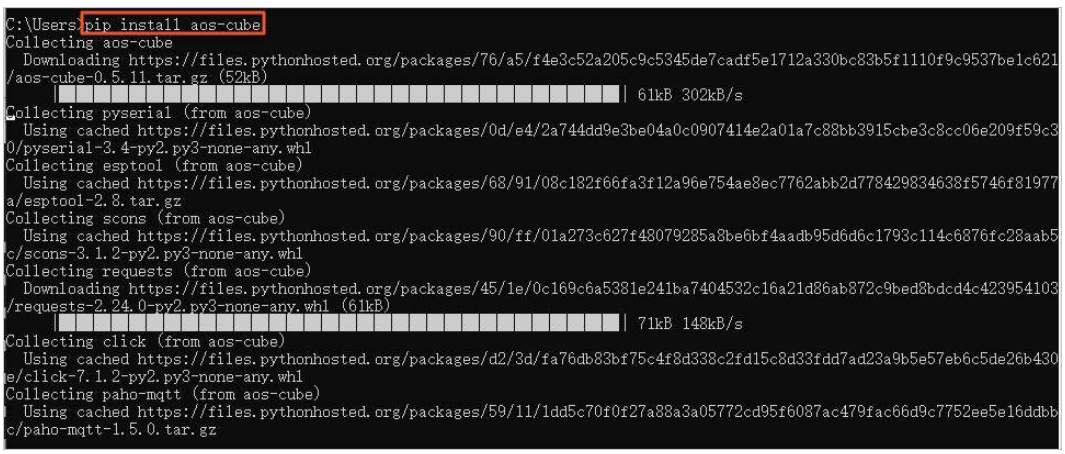
```
python -V
```

输出 Python 版本信息表示 Python 安装成功。

4. 安装 aos-cube。

执行以下命令安装 aos-cube。

```
pip install aos-cube
```



返回类似如下信息，说明安装成功。

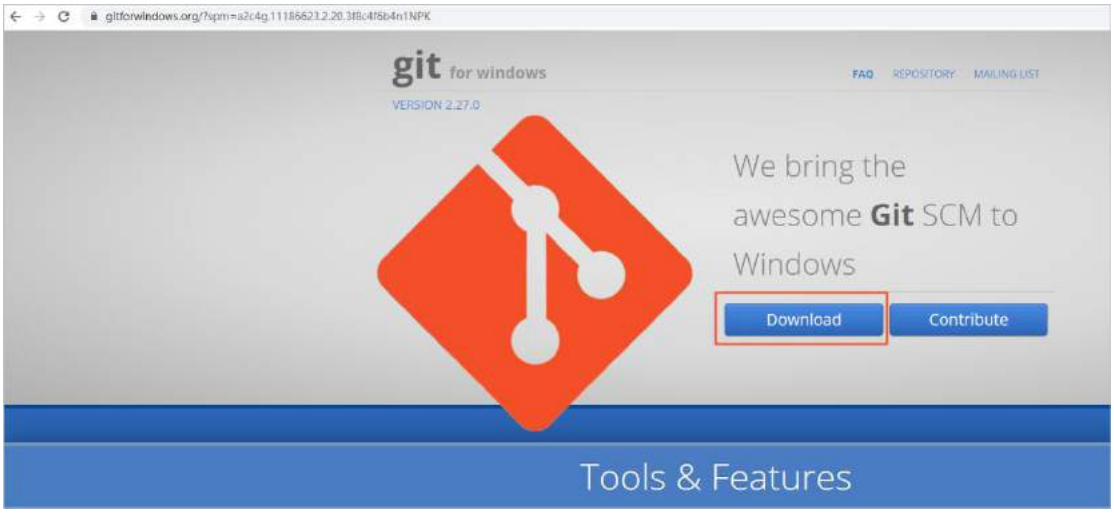
```
Installing collected packages: aos-cube

    Running setup.py install for aos-cube ... done

Successfully installed aos-cube-0.5.11
```

步骤二：安装 Git

1. 下载 Git 安装程序。

    a. 使用浏览器打开 Git 官网。

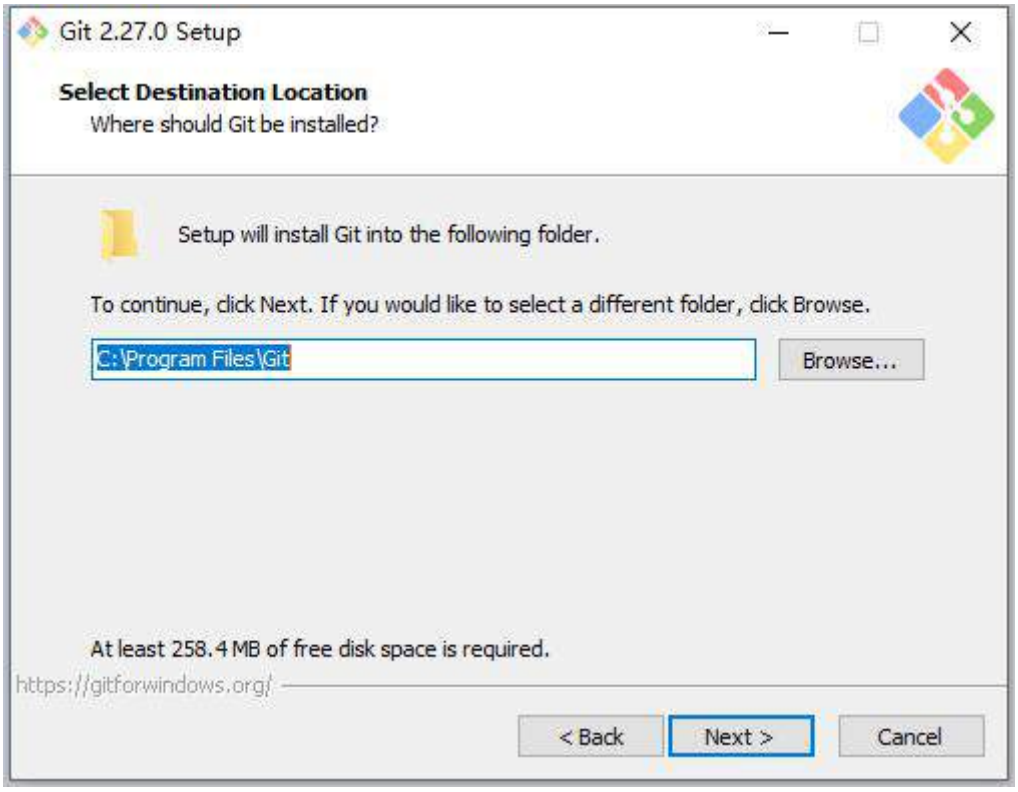
    b. 单击 **Download**。



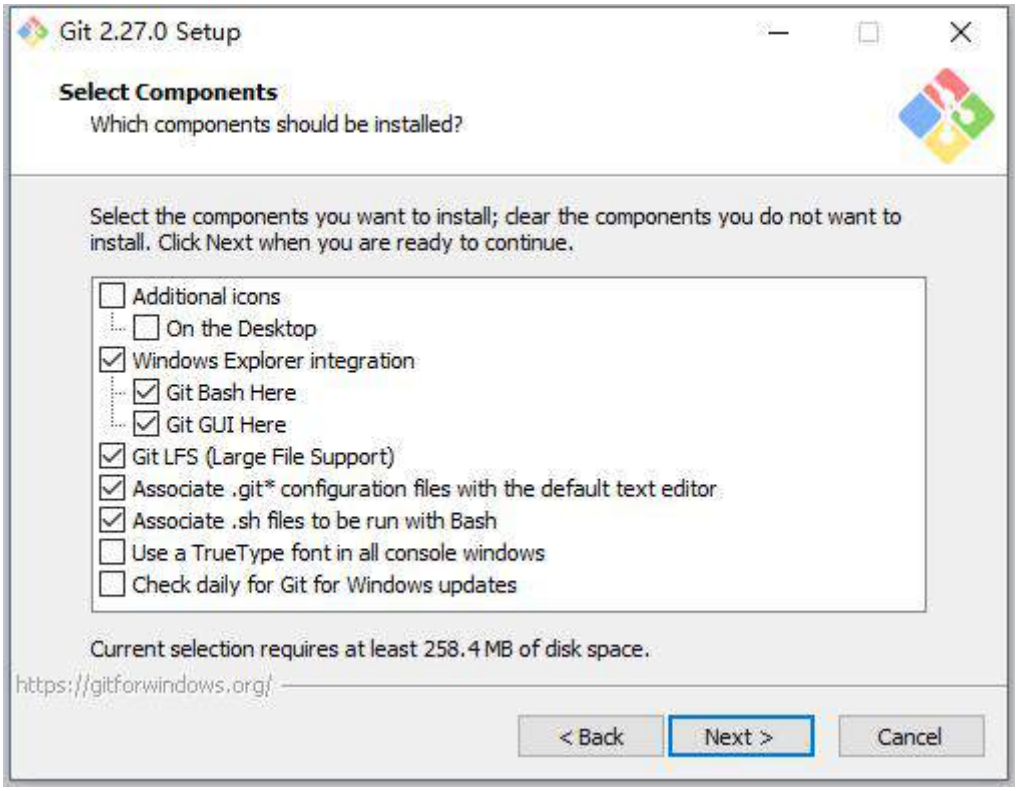
2. 下载完成后，双击 **Git-2.27.0-64-bit .exe**，开始安装 Git。

3. 单击 **Next**。

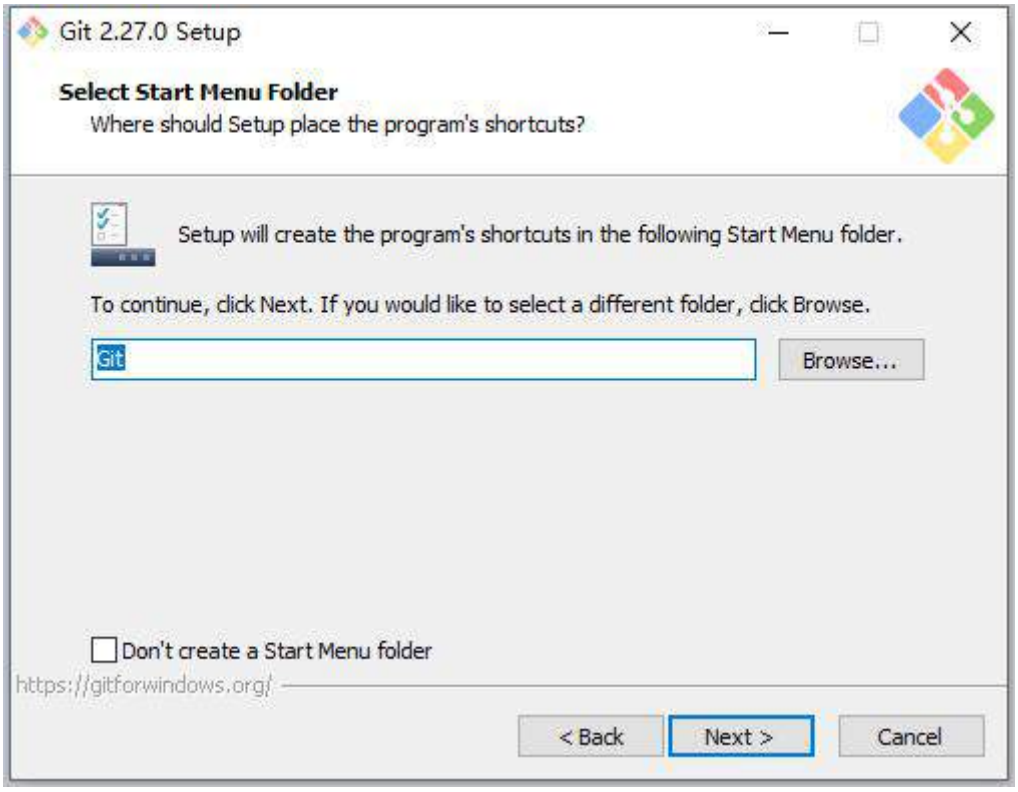
4. 按默认配置，单击 **Next**。



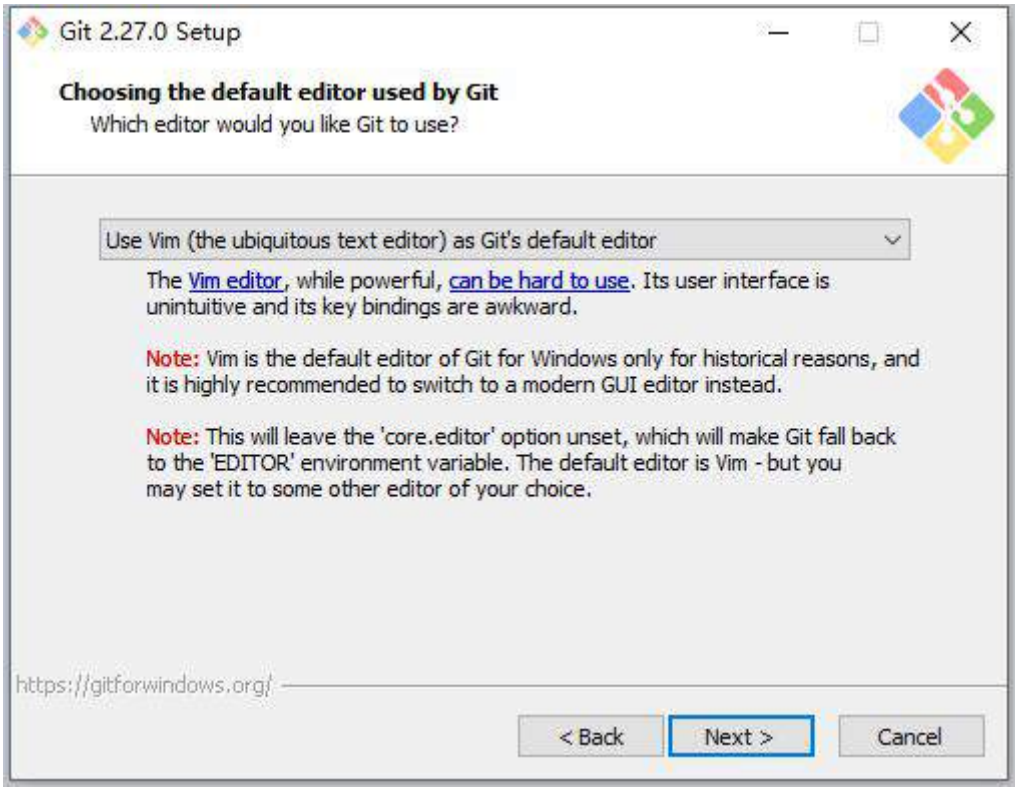
5. 按默认配置，单击 **Next**。



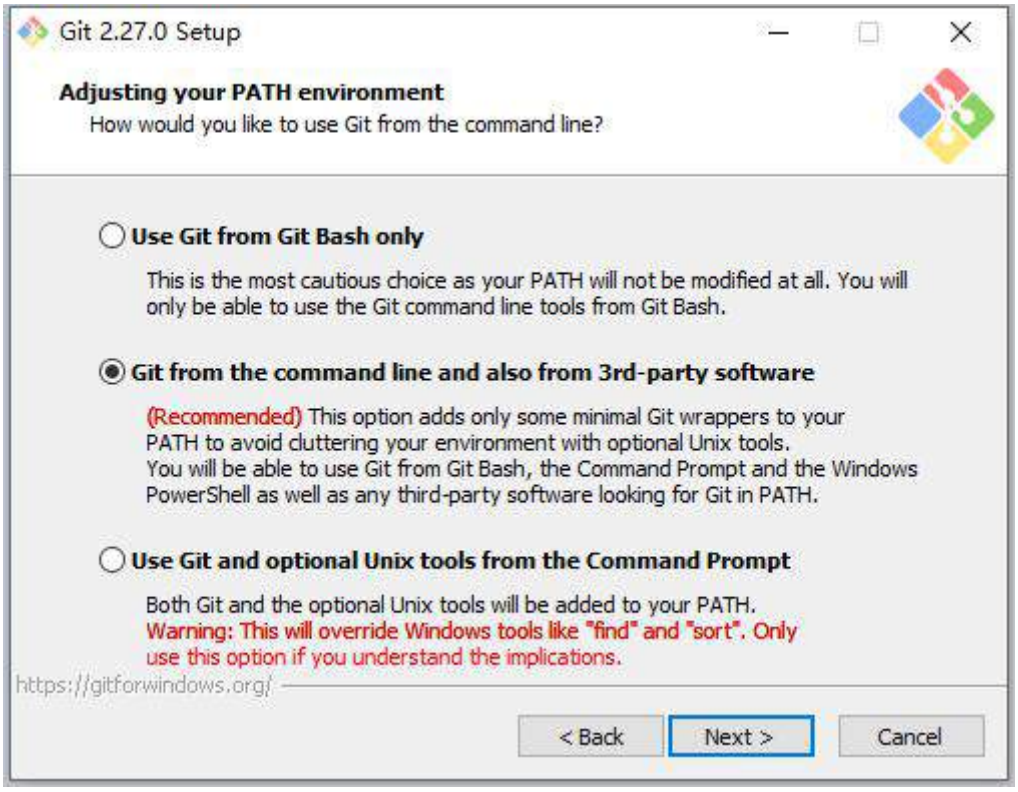
6. 按默认配置，单击 **Next**。



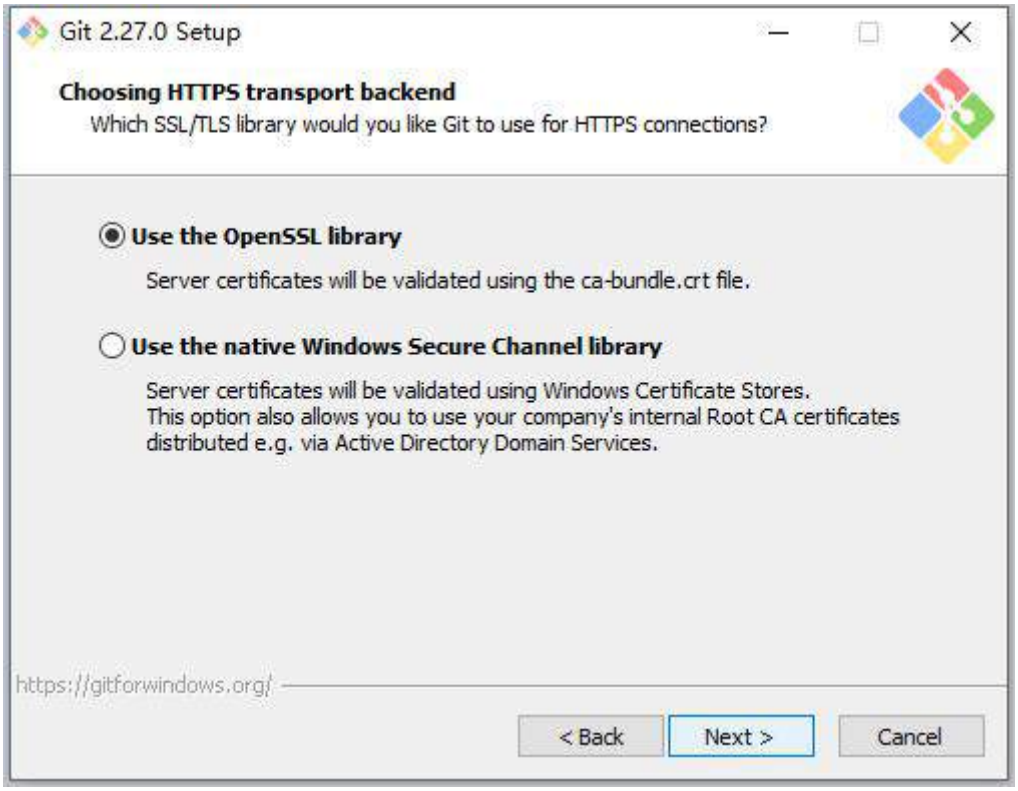
7. 单击 **Next**。



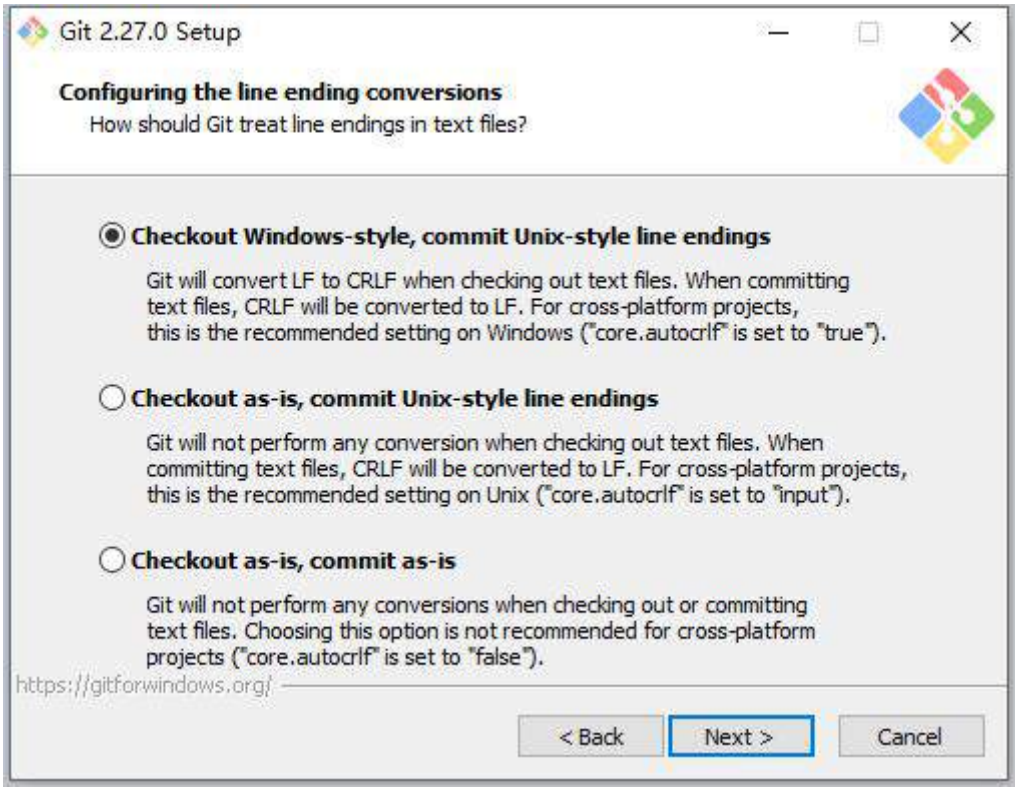
8. 按默认配置，单击 **Next**。



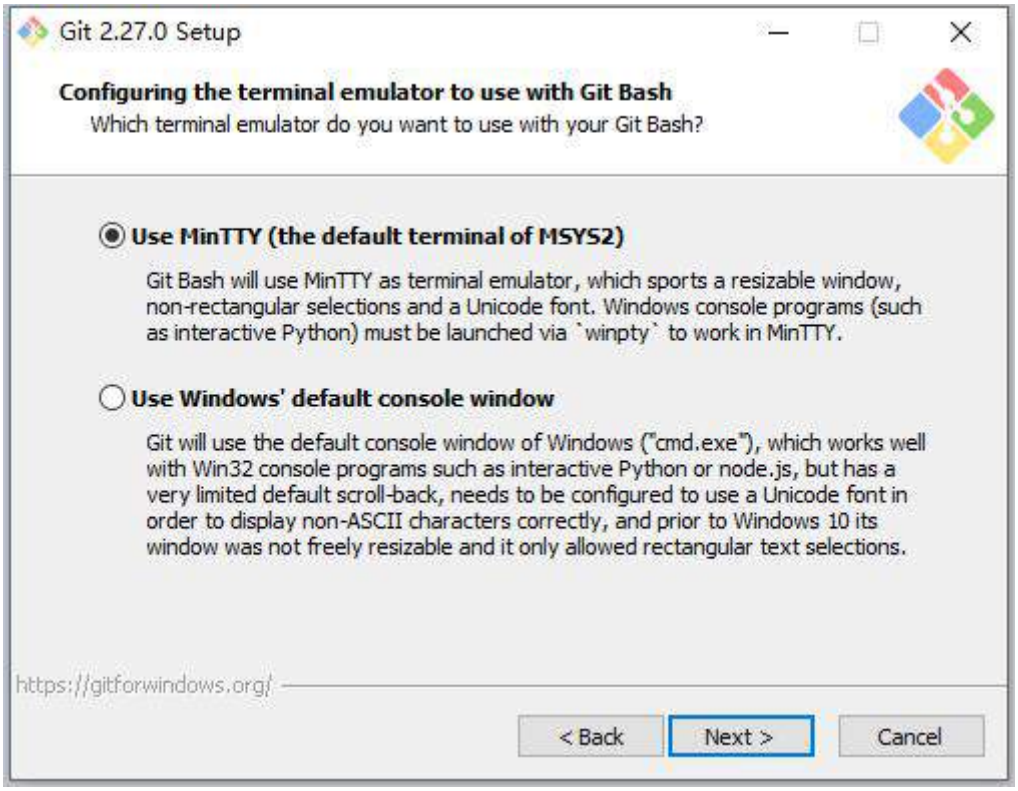
9. 按默认配置，单击 **Next**。



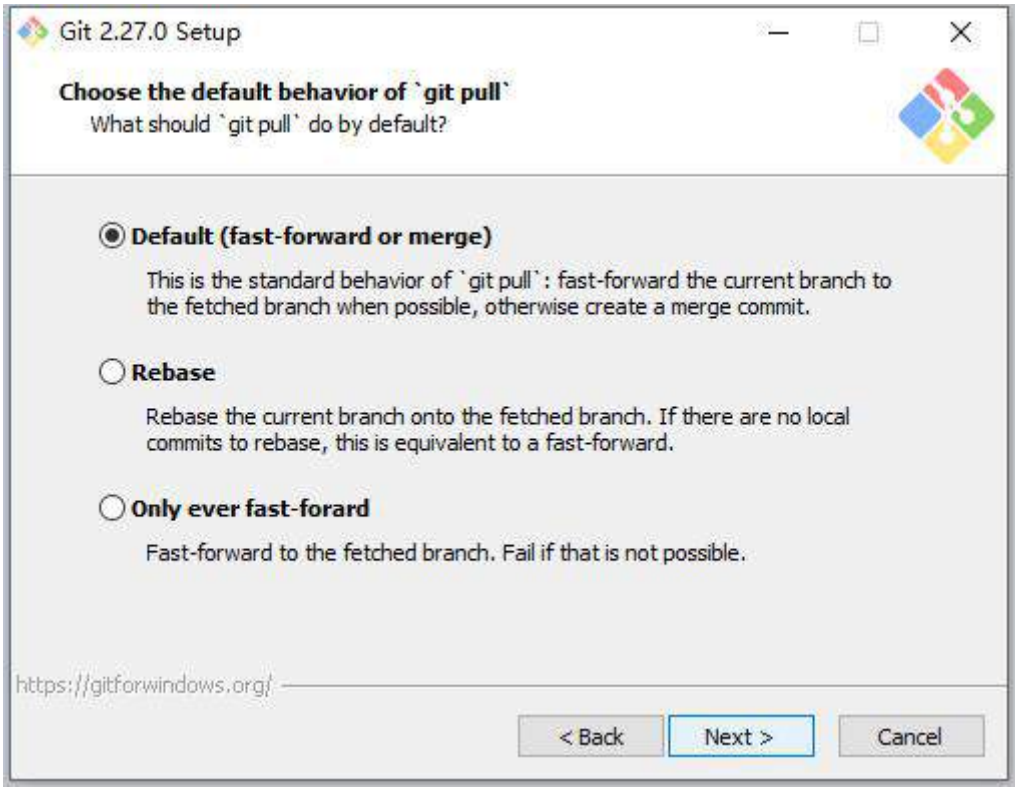
10. 按默认配置，单击 **Next**。



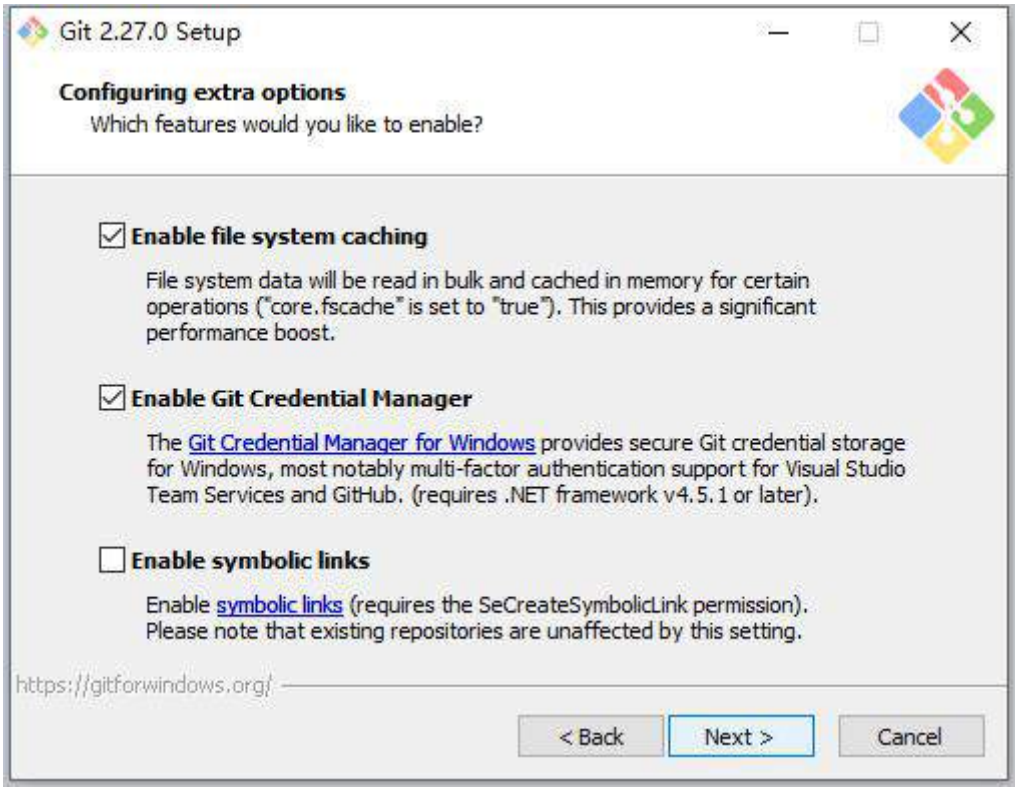
11. 按默认配置，单击 **Next**。



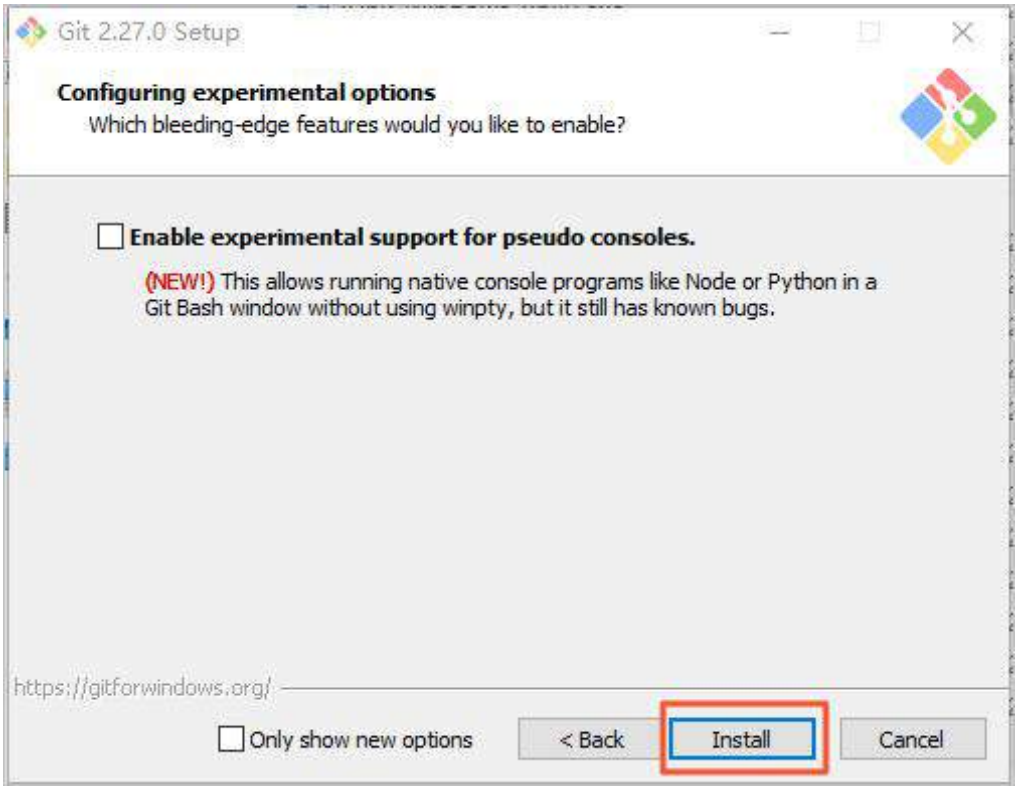
12. 按默认配置，单击 **Next**。



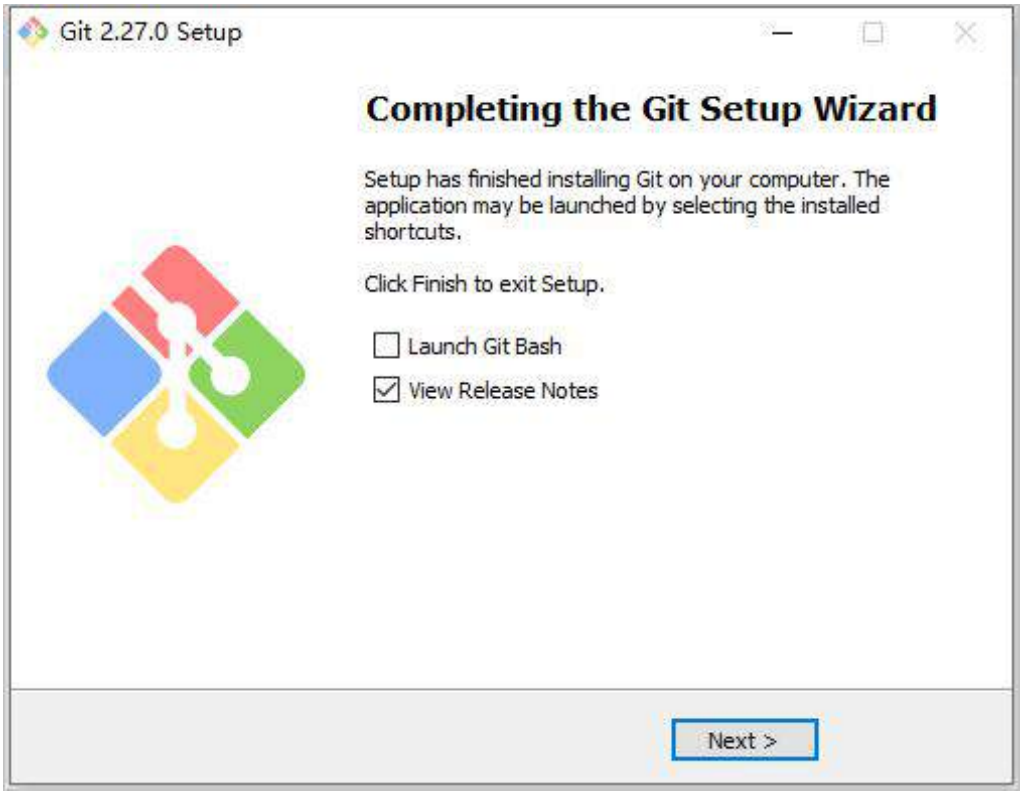
13. 按默认配置，单击 **Next**。



14. 单击 **Install**。



15. 单击 **Next**。



16. 验证 Git 是否安装成功。

打开命令窗口，执行以下命令。输出 Git 版本信息，说明 Git 安装成功。

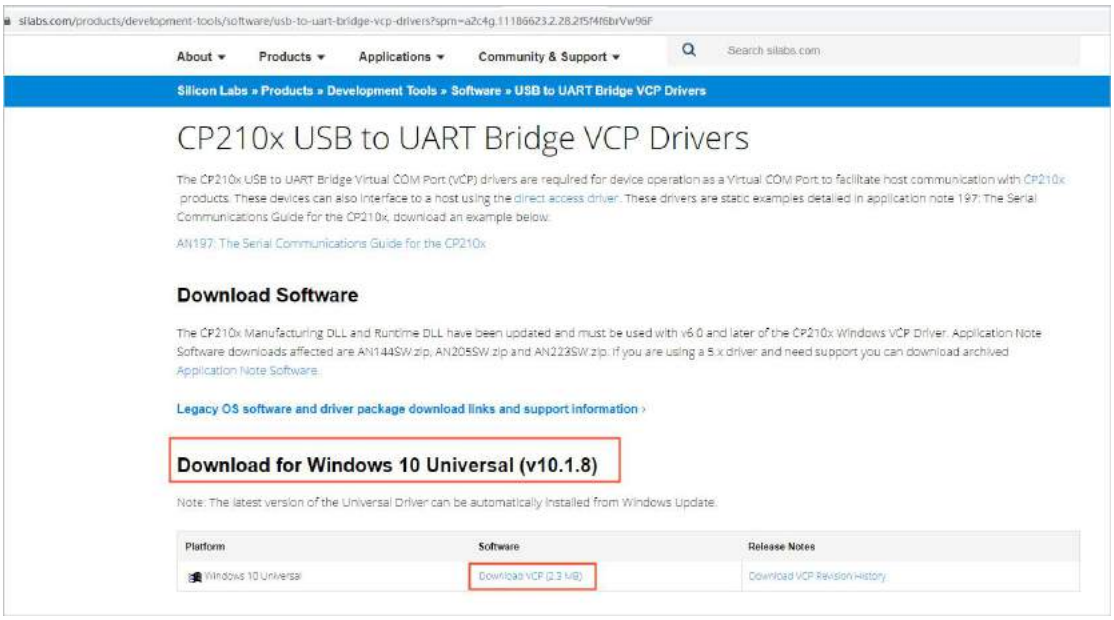
```
git --version
```

步骤三：安装 CP210x 系列驱动

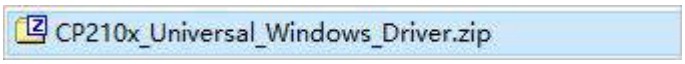
1. 下载 Windows 版的 CP210x 系列驱动。

   a. 打开驱动官网之 CP210x 系列。

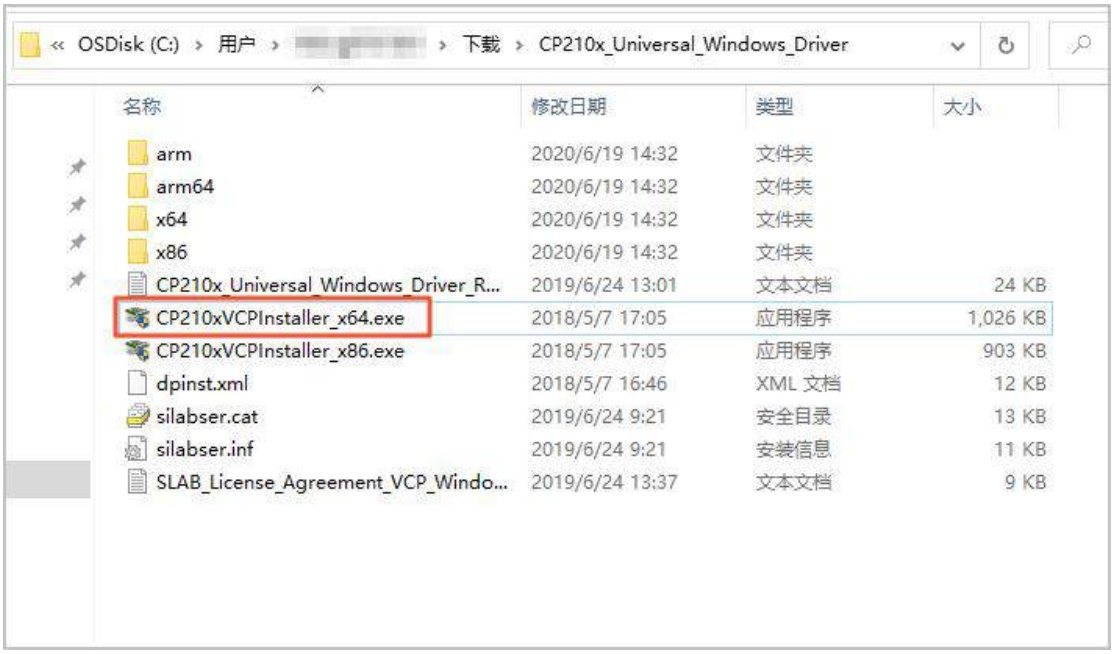
   b. 选择 **Download for Windowss 10 Universal（v10.1.8）**，单击 **Download VCP（2.3MB）**。



2. 解压下载好的文件 **CP210x_Universal_Windows_Driver.zip**。



3. 打开解压后的文件，双击 **CP210xVCPInstaller_x64.exe** 安装程序。



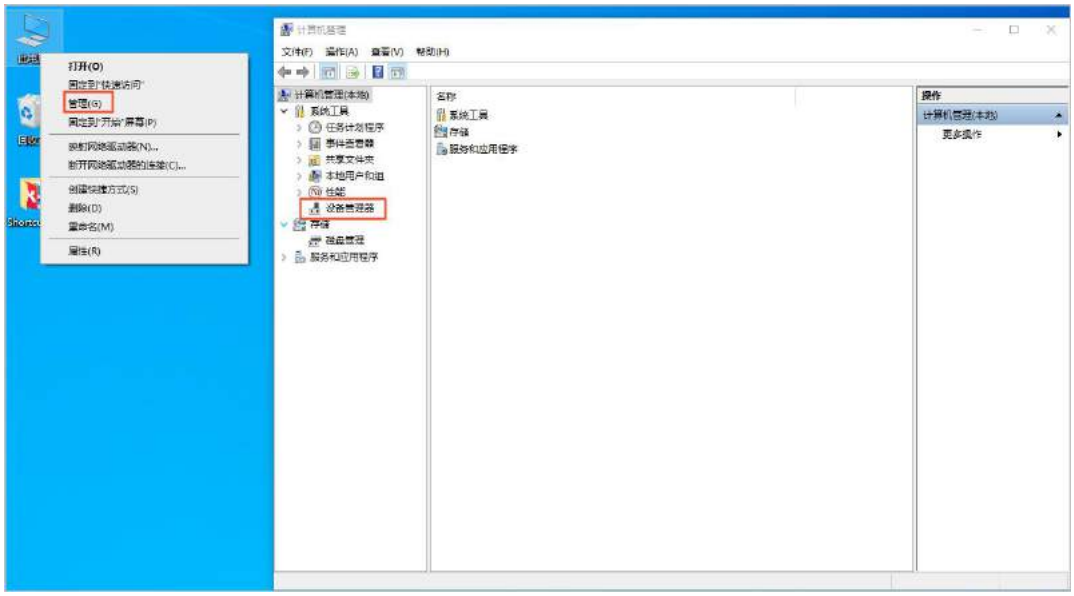
4. 单击**下一步**，开始安装。

5. 单击**完成**，完成 CP210x 系列驱动的安装。

6. 验证 CP210x 系列驱动是否安装成功。

a. 通过 USB 线缆把物联网硬件设备连接到电脑上。

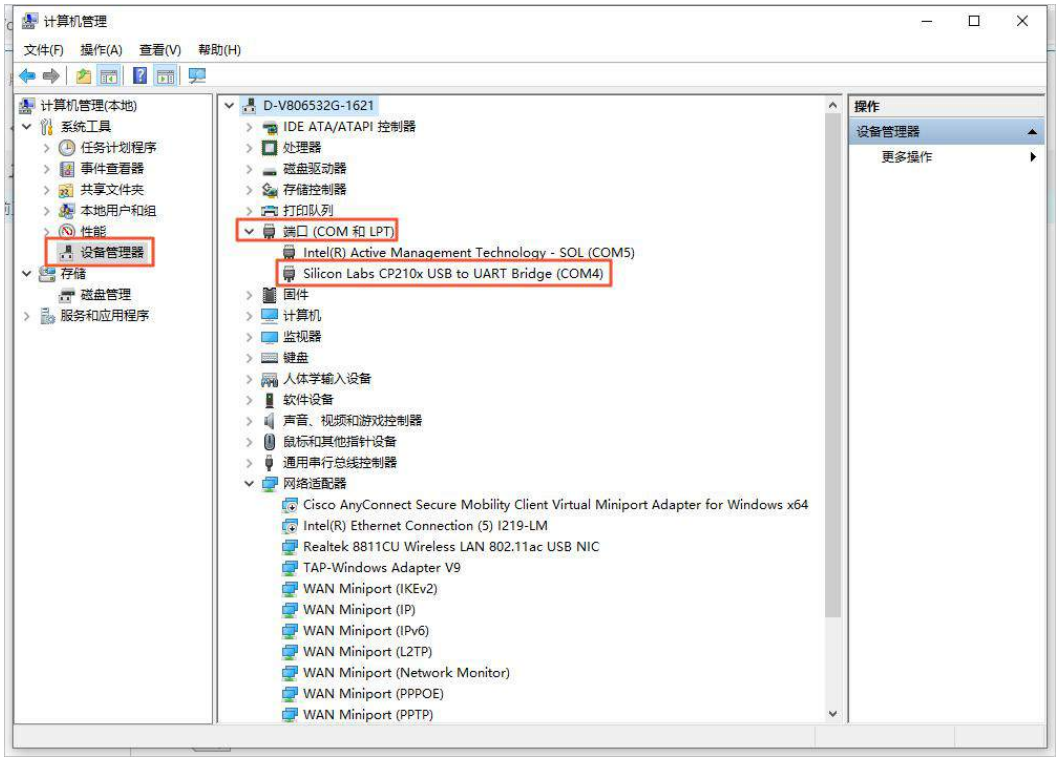
b. 右键单击**此电脑**，单击**管理**。



c. 在**计算机管理**页面，单击设**备管理器 > 端口**。

如果显示类似如下信息，说明 CP210x 系列驱动安装成功。

Silicon Labs CP210x USB to UART Bridge（COM4）

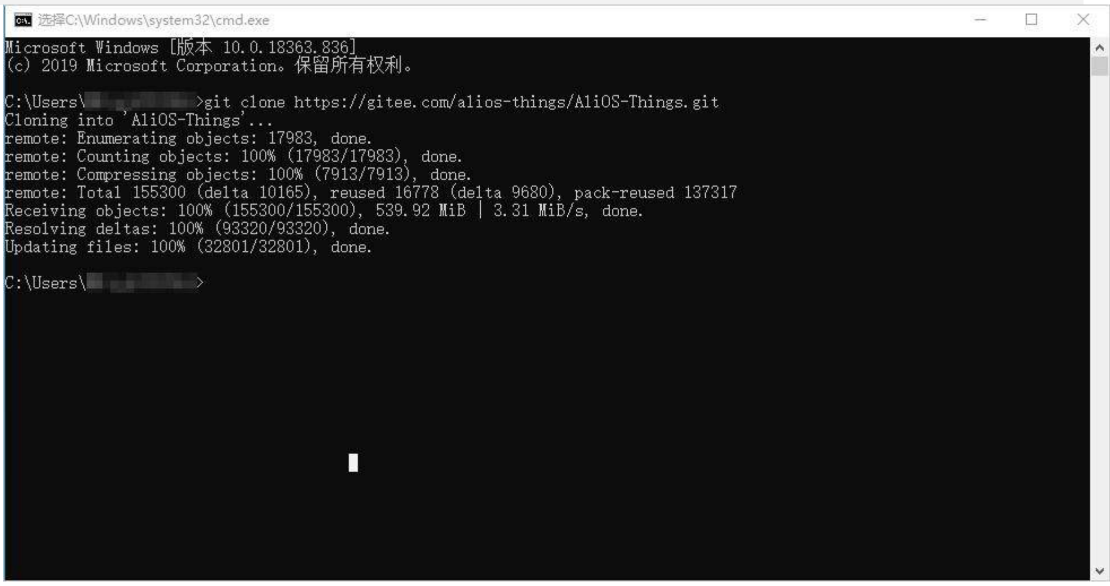


步骤四：获取 AliOS Things 源码

1. 打开命令窗口。
2. 执行以下命令，下载 AliOS Things 源码。

```
git config --global core.compression -1
```

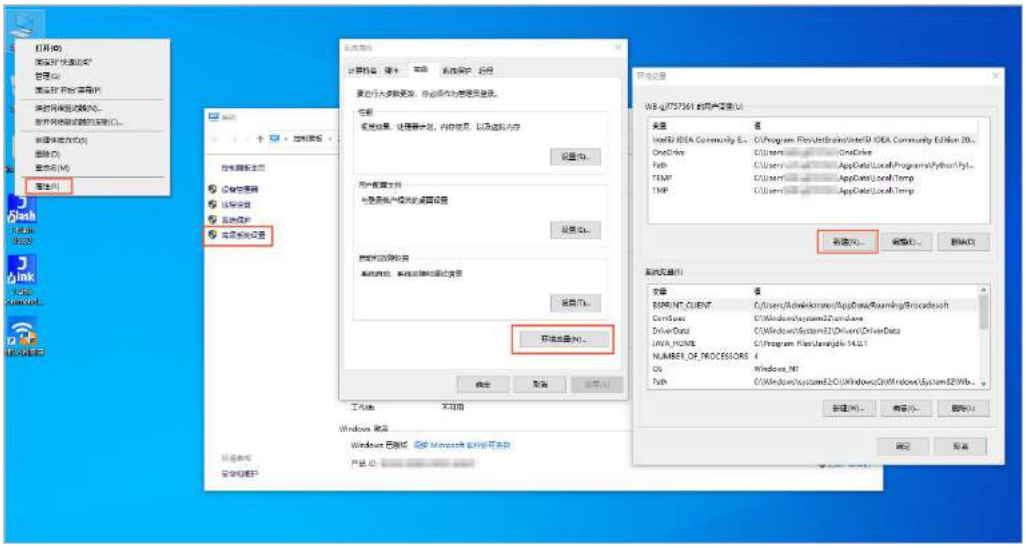
```
git clone -b rel_3.1.0 https://gitee.com/alios-things/AliOS-Things.git
```



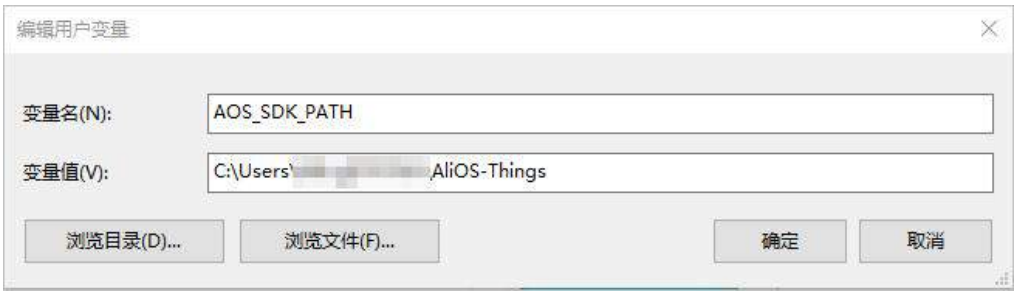
步骤五：配置 AliOS Things 源码环境变量

因为 aos-cube 会根据 AOS_SDK_PATH 环境变量来定位 AliOS Things 源码，所以要配置 AOS_SDK_PATH 环境变量。

1. 单击**此电脑 > 属性 > 高级系统设置 > 环境变量**。

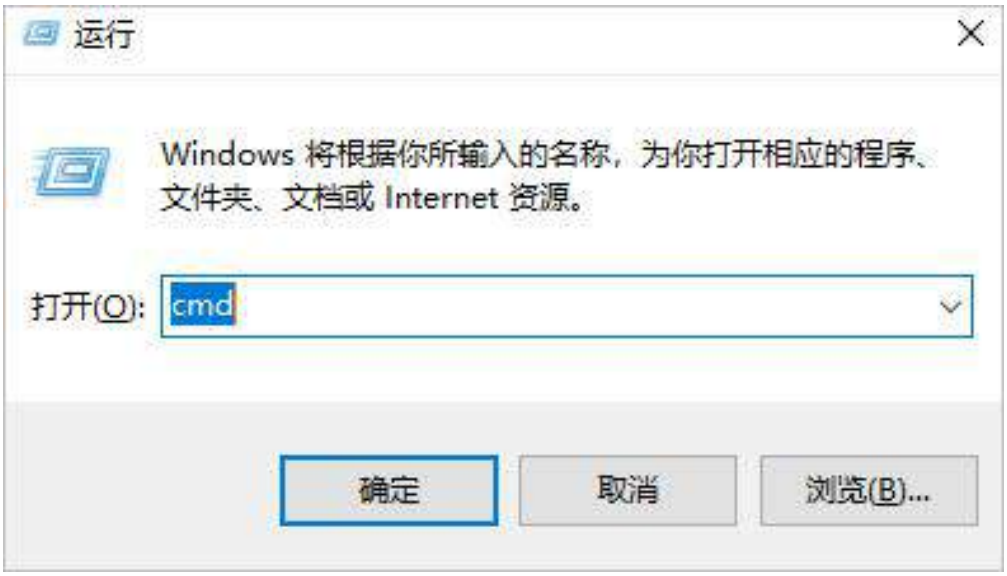


2. 在**环境变量**界面，单击**用户变量**对应的**新建**，在弹出的**编辑用户变量**框里面，变量名输入 AOS_SDK_PATH，变量值输入 AliOS Things 源码所在的目录，单击**确定**退出。



3. 查看环境变量是否生效。

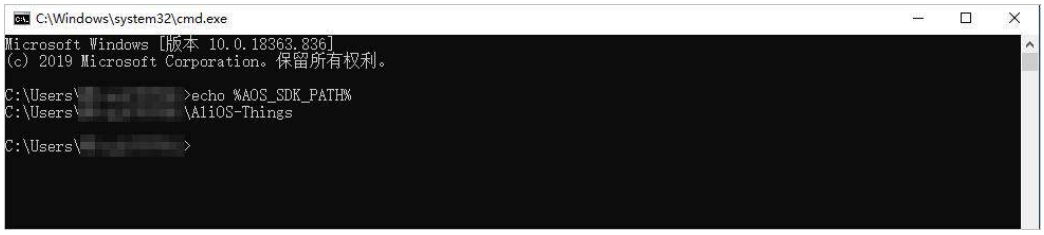
   a. 使用快捷键 **Win+R**，在对话框里输入 **cmd**，按 **Enter** 键，打开命令窗口。



   b. 执行以下命令，查看 AOS_SDK_PATH 环境变量。

```
echo %AOS_SDK_PATH%
```

如果返回 AliOS Things 源码的目录，表示 AOS_SDK_PATH 环境变量配置成功。



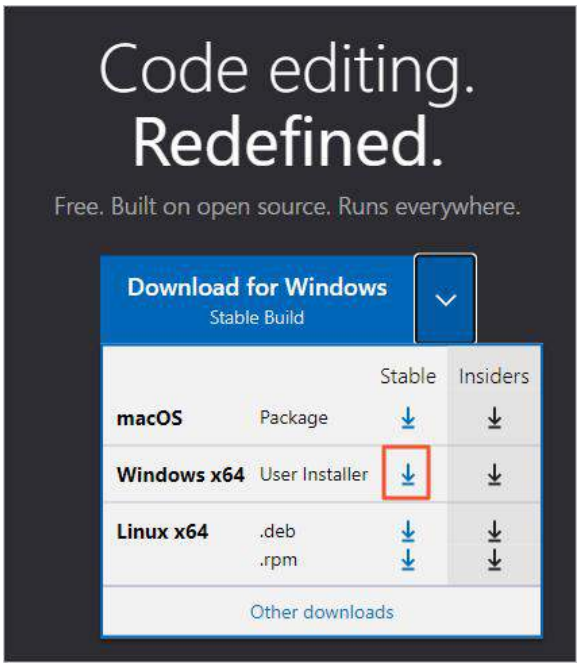
步骤六：安装 Visual Studio Code

1. 下载 VSCode 安装程序。

   a. 打开 VSCode 官网。

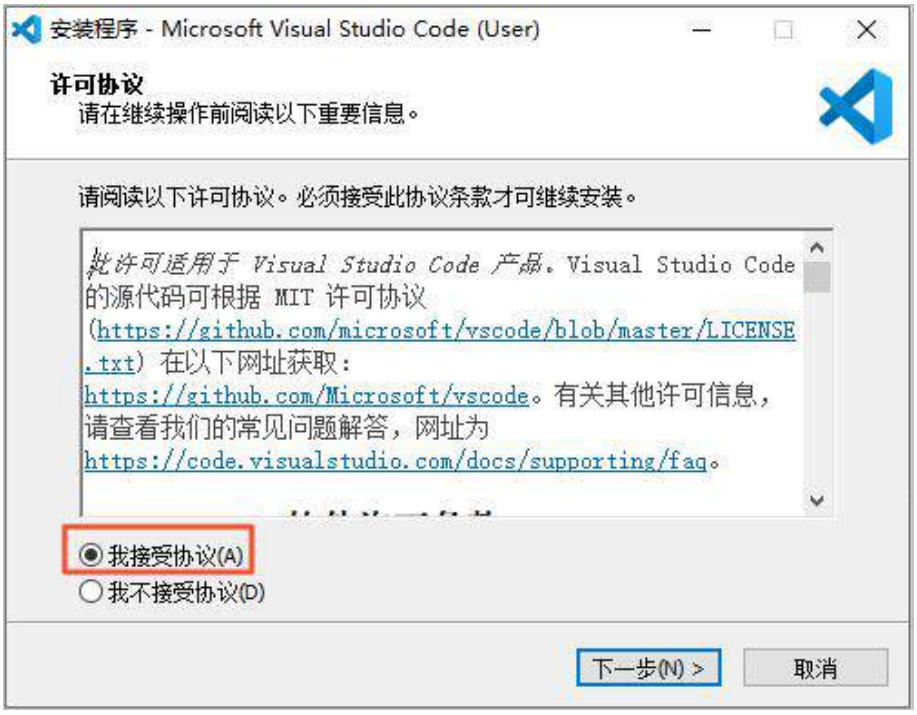
   b. 单击下图的下拉图标。

   c. 单击 **Windows x64** 平台的**下载**图标。

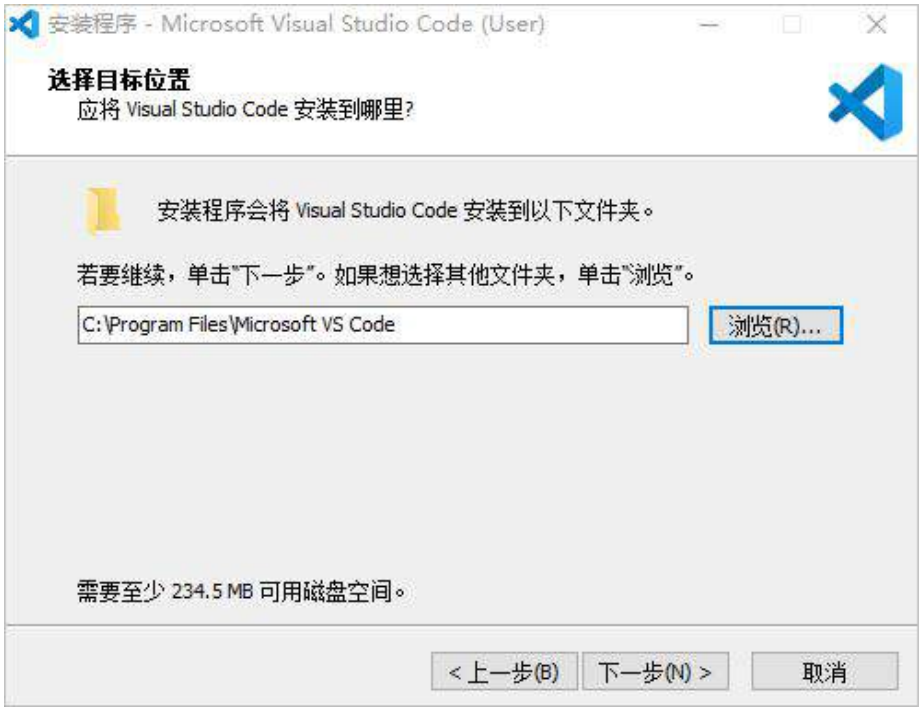


2. 双击 **VSCodeUserSetup-x64-1.45.1.exe** 开始安装。

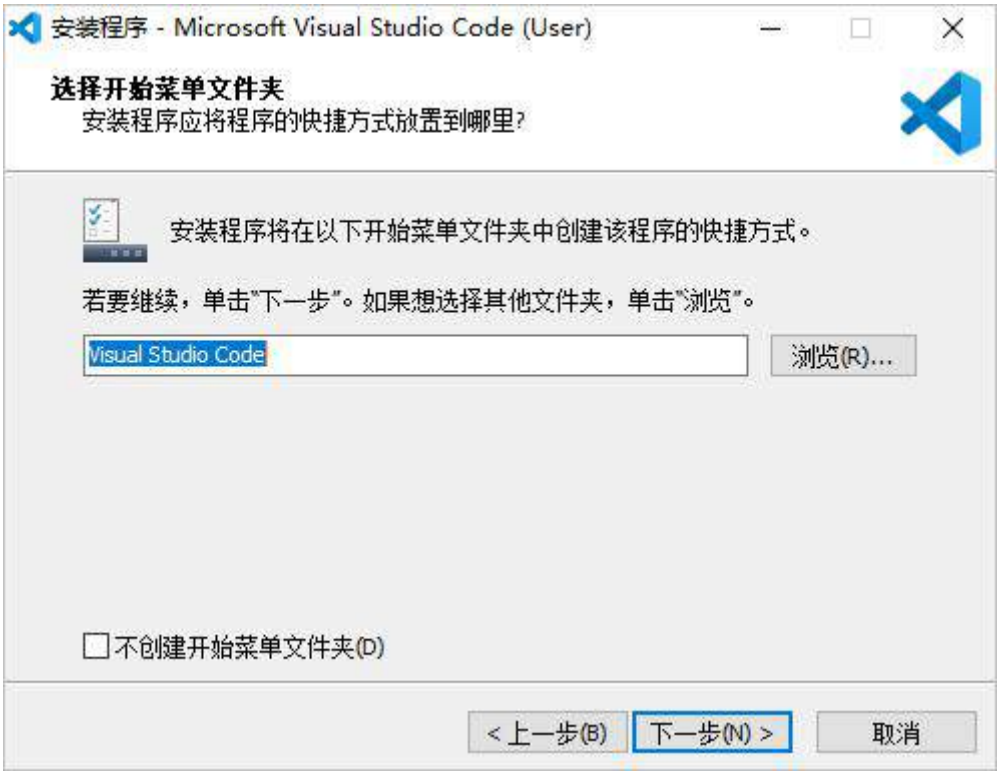
3. 勾选**我接受协议（A）**，单击下一步。



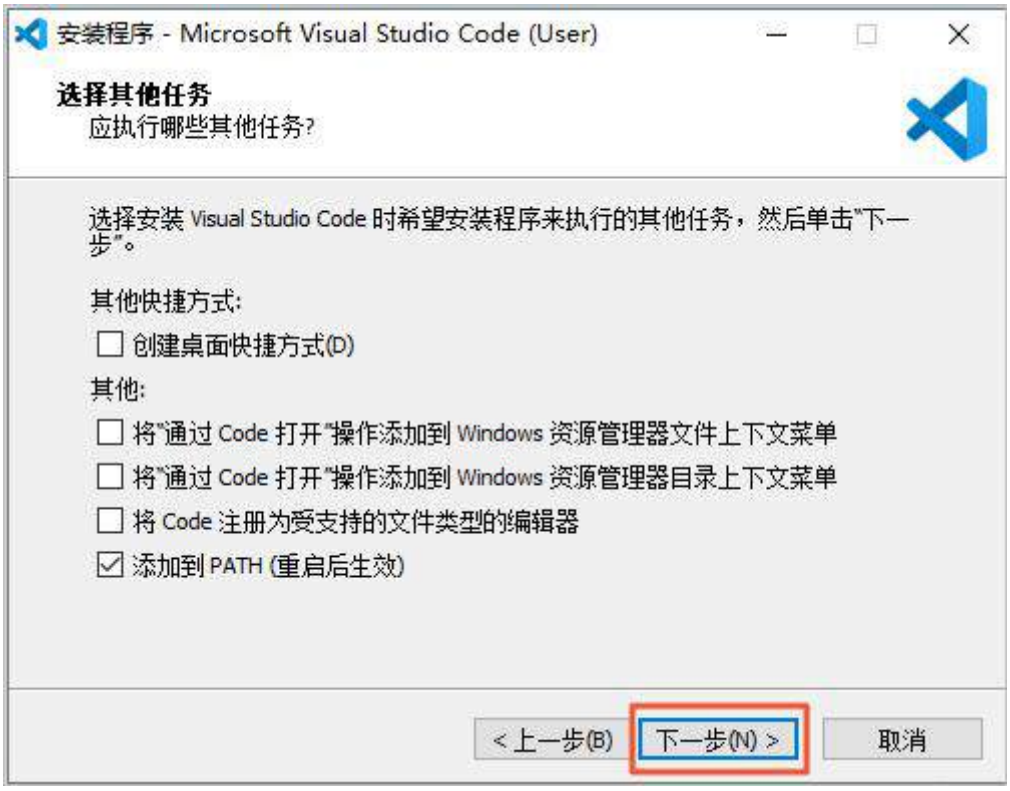
4. 选择安装目录，单击**下一步**。



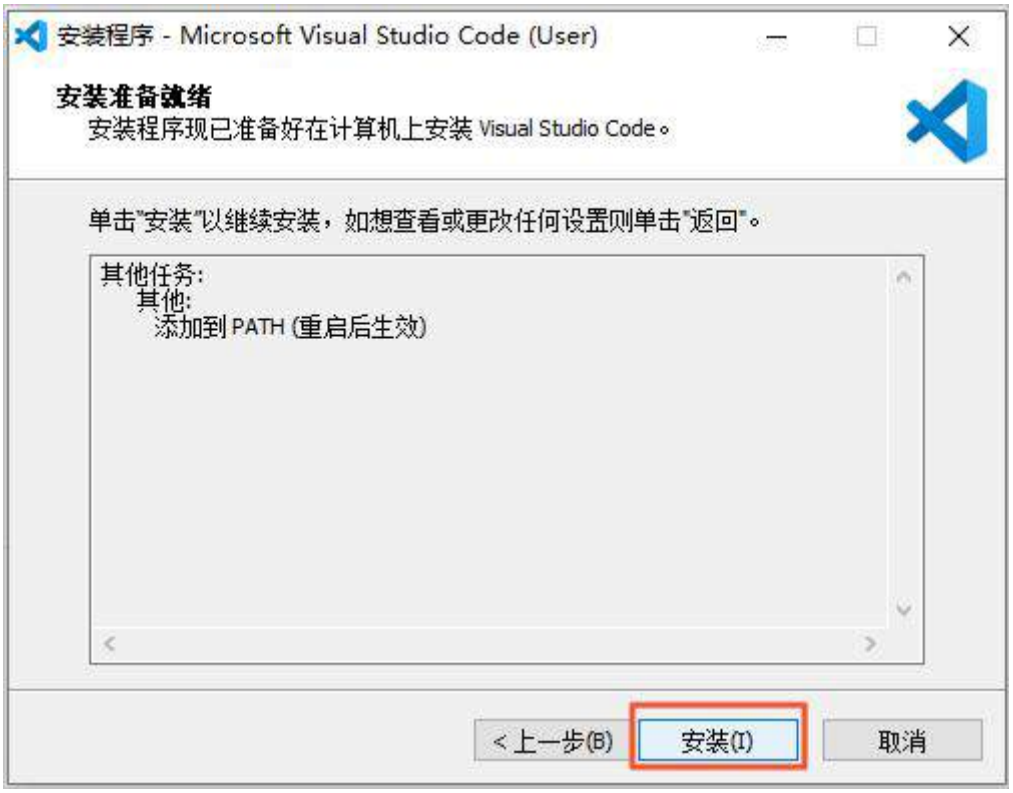
5. 按默认配置，单击**下一步**。



6. 按默认配置，单击**下一步**。



7. **单击安装。**

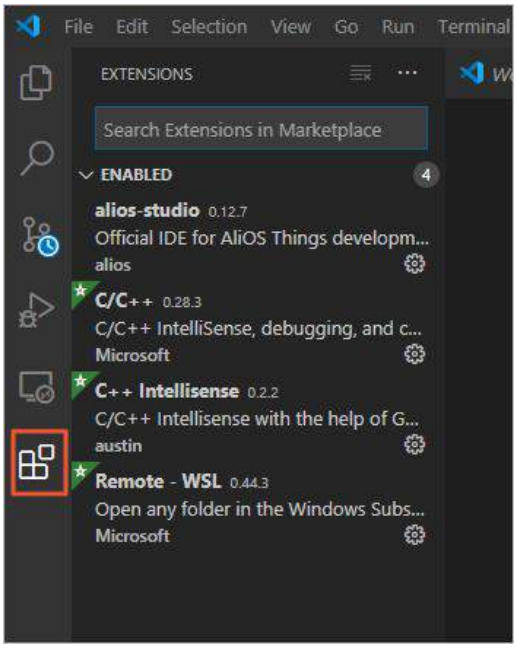


8. **单击完成，完成安装。**

步骤七：安装 alios-studio 插件

操作步骤：

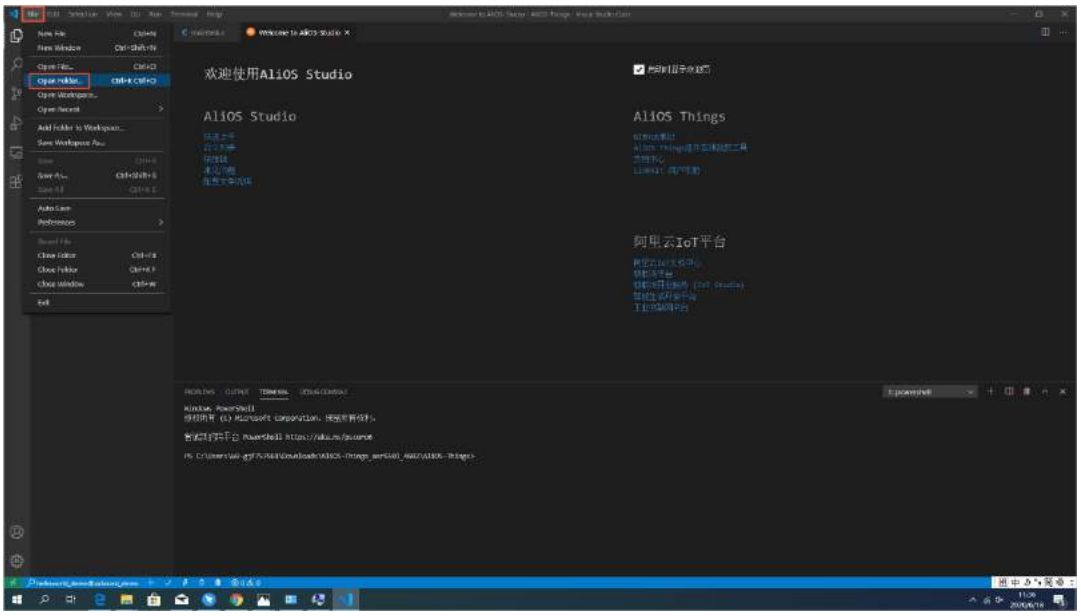
1. 打开 VSCode，单击左侧活动栏的 **Extensions**  。



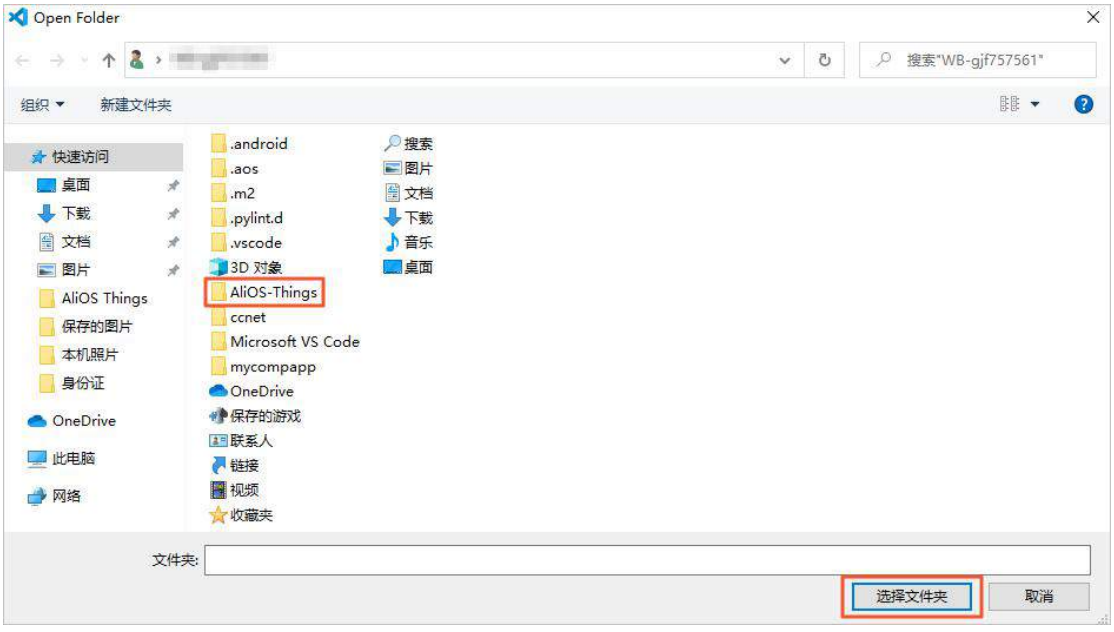
2. 在搜索框里输入 alios-studio，单击 **Install**。

步骤八：使用 VSCode 导入 AliOS Things 源码

1. 打开 VSCode，单击 **File > Open Folder**。



2. 选择源码所在的文件夹，单击**选择文件夹**，导入源码。



二、AliOS Things 代码目录结构

AliOS Things core SDK 包含以下目录。

```
AliOS-Things
├── application
```

```
│   ├── example              # 示例代码
│   └── profile              # 典型场景的应用方案
├── build                    # 编译构建相关工具和脚本
├── components               # 功能组件
│   ├── dm                   # 设备管理组件
│   │   ├── bootloader
│   │   ├── ota
│   │   ├── ulog
│   │   └── und
│   ├── linkkit              # 阿里云 IoT 连接套件
│   ├── network              # IP 网络协议栈组件
│   │   ├── http
│   │   ├── lwip
│   │   └── netmgr
│   ├── security             # 安全类组件
│   │   └── mbedtls
│   └── utility              # 工具类组件
│       ├── cjson
│       └── yloop
├── core                     # 内核及相关组件
├── document                 # 说明文档
├── include                  # 组件对外的头文件
├── platform                 # 芯片平台支持和 BSP
│   ├── arch                 # 架构移植
│   ├── board                # 板级支持
│   └── mcu                  # MCU, SoC 移植支持
└── projects                 # 为不同开发环境提供的工程相关文件
```

增值类组件包含以下目录。

```
components
├─── bus                          # 本地通讯协议
│    ├─── canopen
│    ├─── knx
│    ├─── mbmaster
│    └─── usb
├─── dm                           # 设备管理
│    └─── uagent
├─── fs                           # 文件系统
│    ├─── cramfs
│    ├─── fatfs
│    ├─── jffs2
│    ├─── ramfs
│    ├─── spiffs
│    ├─── uffs
│    └─── yaffs2
├─── gui                          # 人机交互界面
│    ├─── freetype-2.5.3
│    └─── littlevGL
├─── language                     # 脚本引擎
│    ├─── jsengine
│    └─── micropython
├─── network                      # IP 网络协议栈
│    ├─── coap
│    ├─── httpdns
│    ├─── libsrtp
│    ├─── lwm2m
```

```
│   ├── mal
│   ├── rtp
│   ├── sal
│   ├── umesh2
│   └── websocket
├── peripherals                    # 外设驱动
│   ├── iot_comm_module
│   │   ├── mal
│   │   └── sal
│   └── sensor
├── security                       # 安全
│   └── linksecurity
├── service                        # 应用组件
│   ├── uai
│   ├── udata
│   └── ulocation
├── utility                # 工具类
│   ├── at
│   ├── debug_tools
│   └── zlib
└── wireless                       # 无线类
    ├── bluetooth
    └── lorawan
```

使用线下开发板进行开发调试

本文主要介绍如何使用 VSCode 进行项目编译、固件烧录及串口监控。

一、前提条件

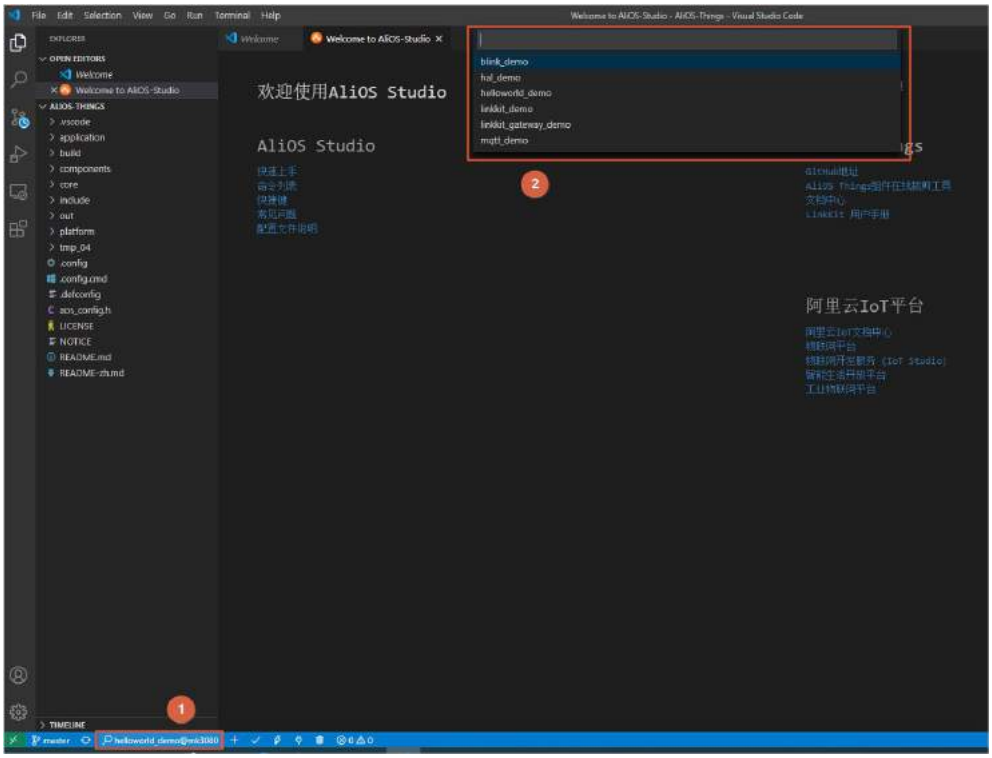
- 完成全部的环境准备工作，具体参见**开发前准备**。
- 准备一个开发板-物联网硬件设备。例如：WiFi esp8266+，端口 Silicon Labs CP210x USB to UART Bridge（COM4）。

步骤一：项目编译

操作步骤：

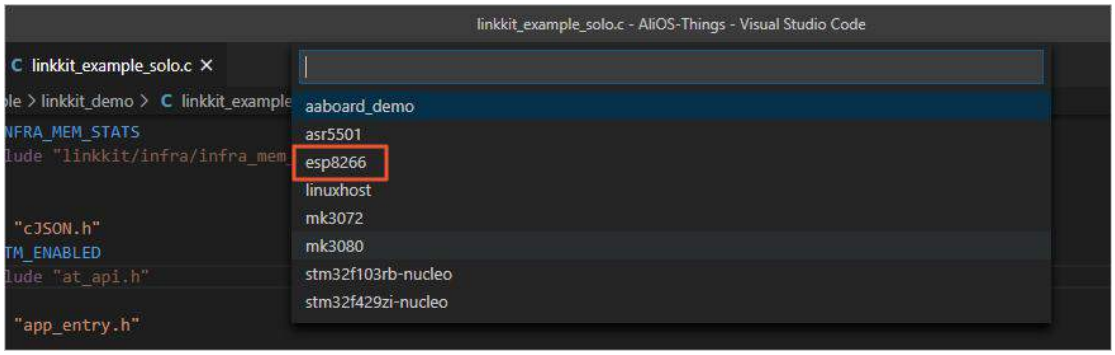
1. 选择应用。

打开 VSCode，单击工具栏项目（图示中①），编辑栏出现所有的应用列表，选择 **helloworld_demo** 项目。



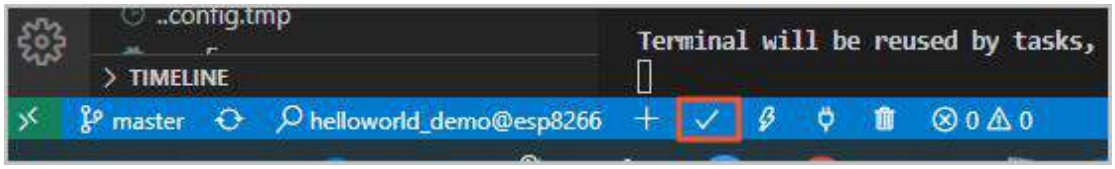
2. 选择开发板。

编辑栏中出现所有的开发板列表，选择 **esp8266**。



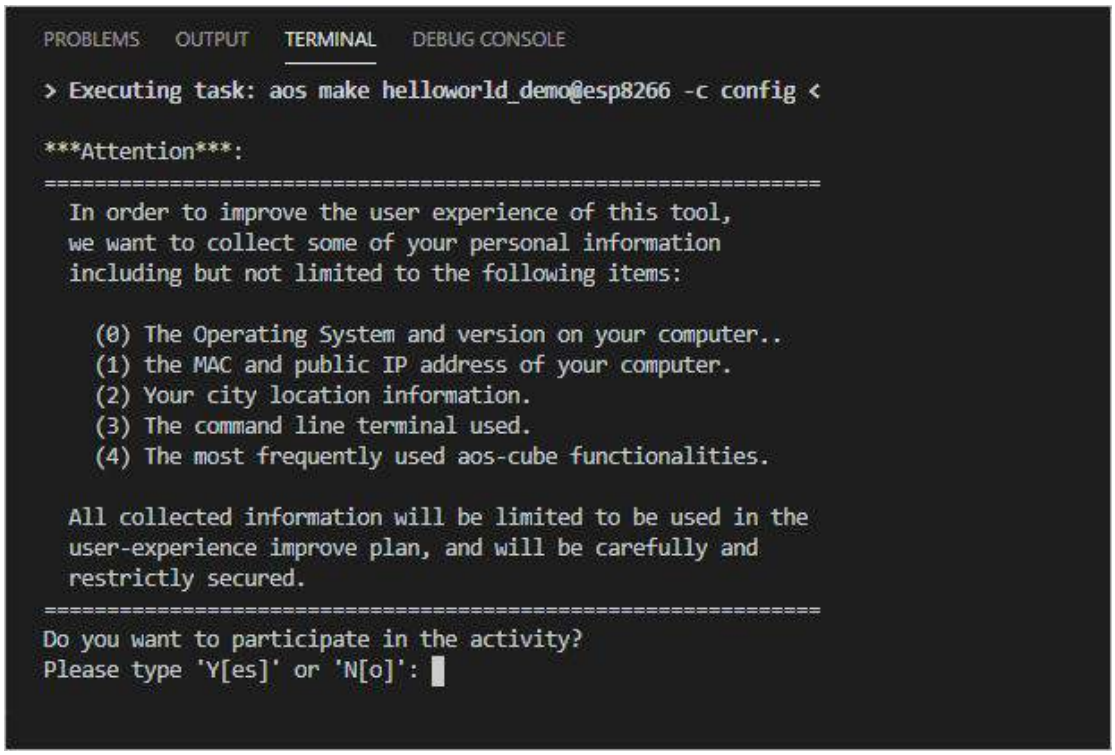
3. 编译。

a. 单击下方工具栏的编译图标  。

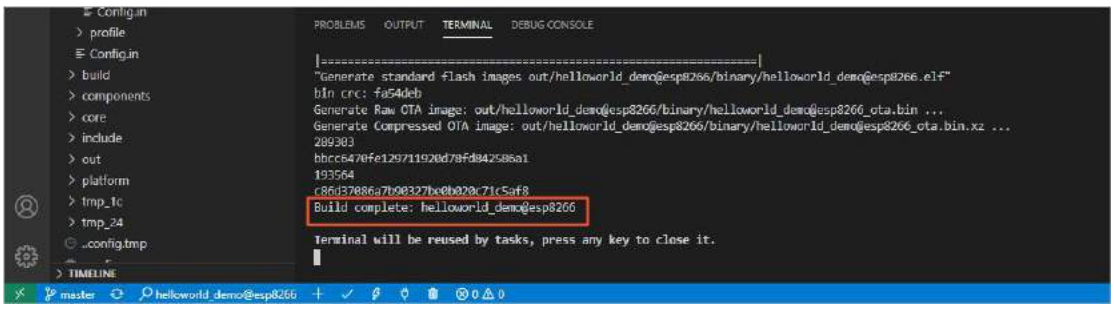


说明：编译过程大概会持续 5 分钟，请耐心等待。

b. 输入 Y，然后按 **Enter** 键。



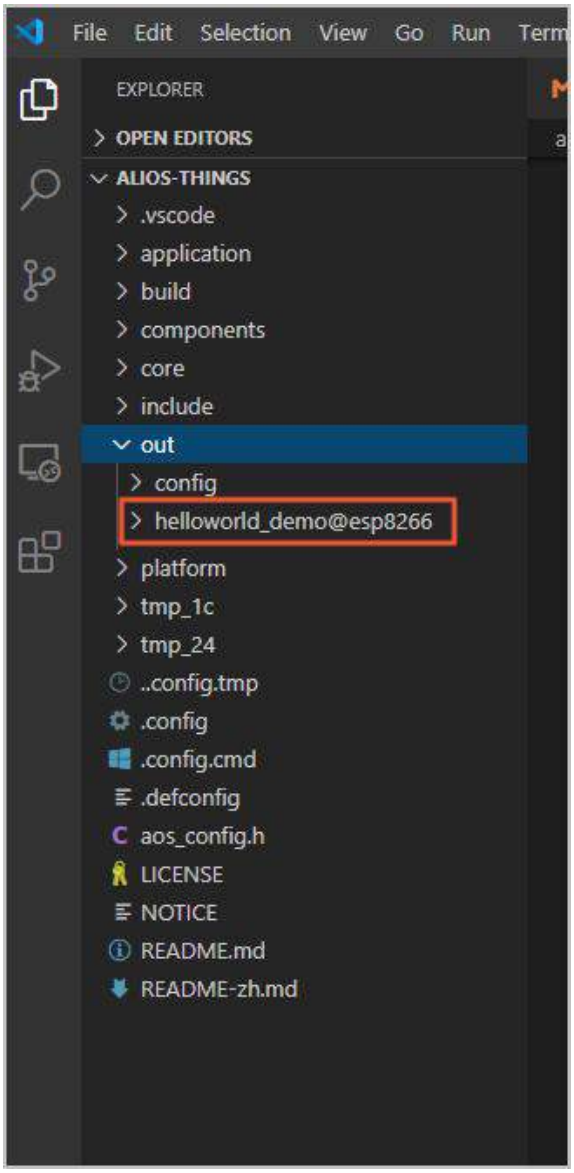
系统显示 Build complete:helloworld_demo@esp8266，表示编译完成。



4. 查看编译后的文件。

单击 **ALIOS-THINGS > out**，您可以看到编译后的文件：

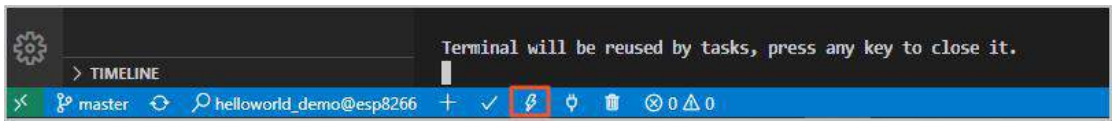
helloworld_demo@esp8266



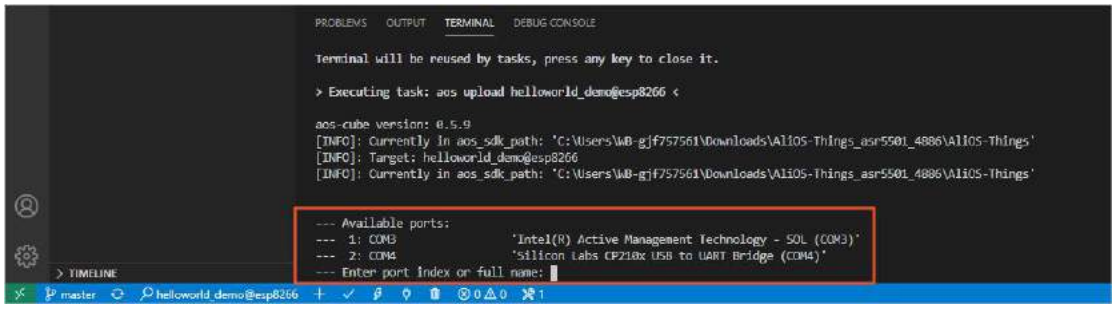
步骤二：固件烧录

操作步骤：

1. 通过 USB Micro 线缆连接好开发板和电脑。
2. 烧录。
   a. 单击下方工具栏的**烧录**图标  。

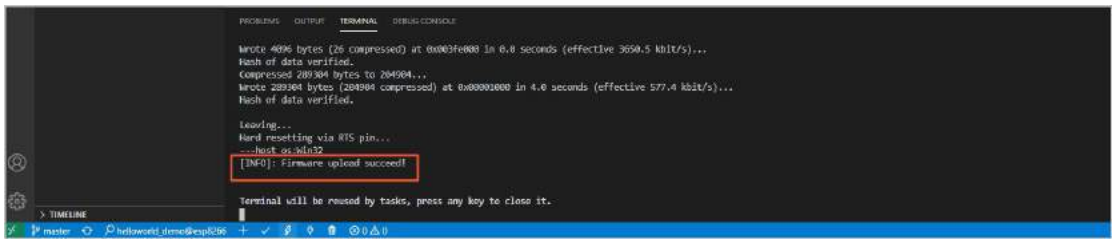


   b. 输入 2，按 **Enter** 键。



说明：此处根据开发板所属的端口进行选择。例如：本文所用的开发板 WiFi-ESP8266+的端口是 Silicon Labs CP210x USB to UART Bridge（COM4），对应的编号为 2，所以输入 2。

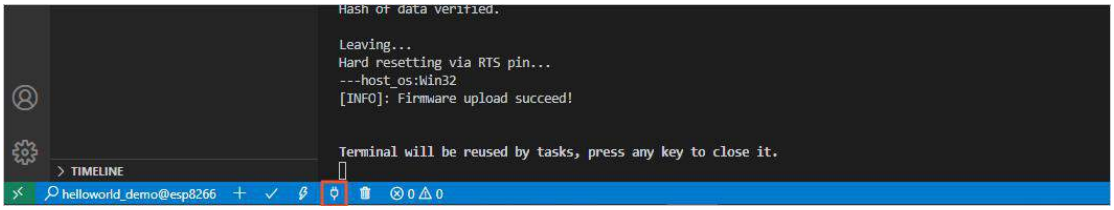
系统显示 Firmware upload succeed!，说明烧录完成。



说明：如果您手上没有可用的开发板，也可以使用**线上的开发板做开发调试**。

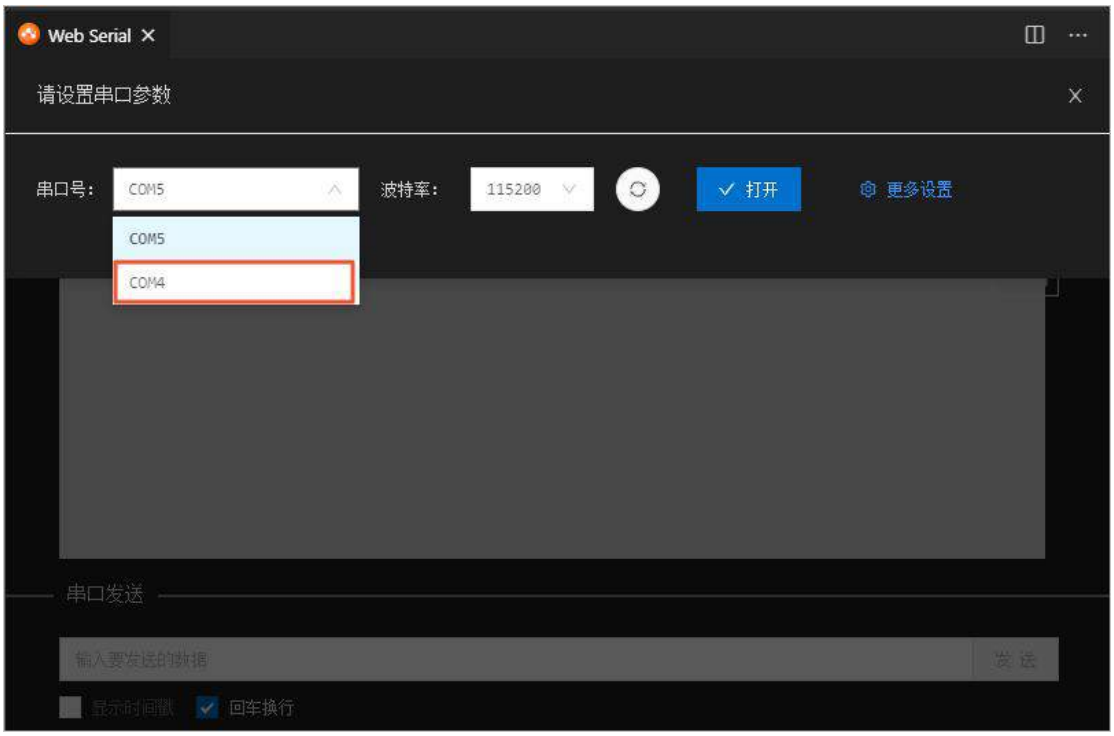
步骤三：串口监控

1. 单击下方工具栏的串口监控图标  。

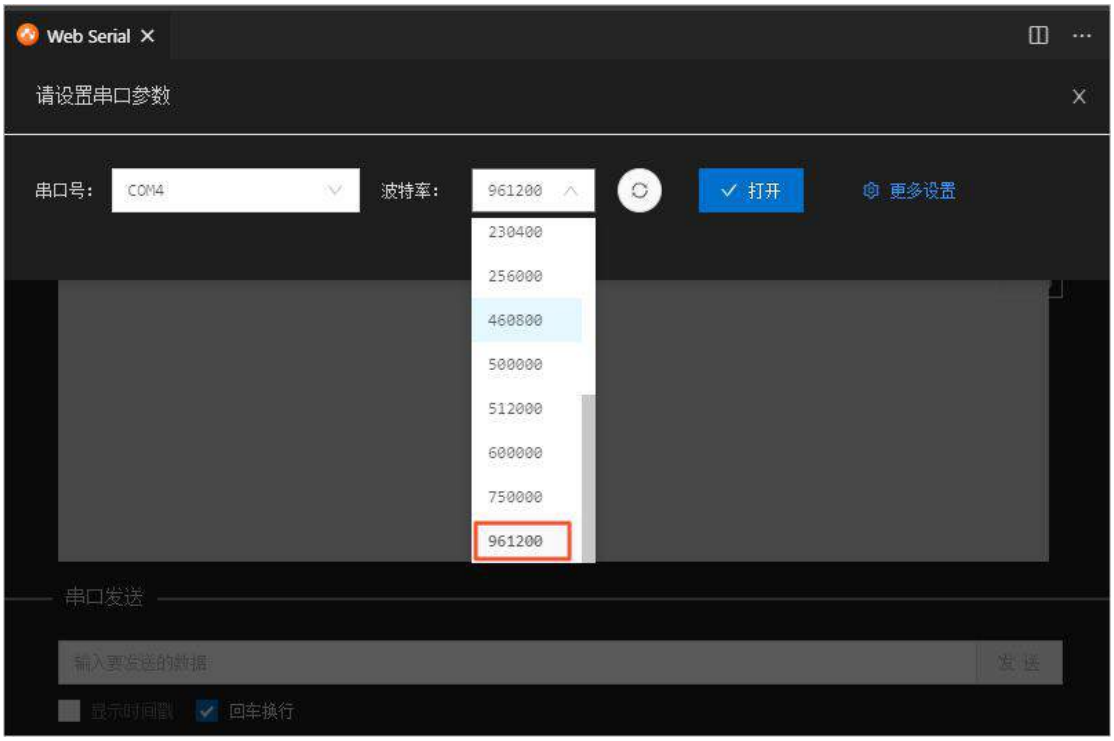


2. **设置串口号和波特率。**

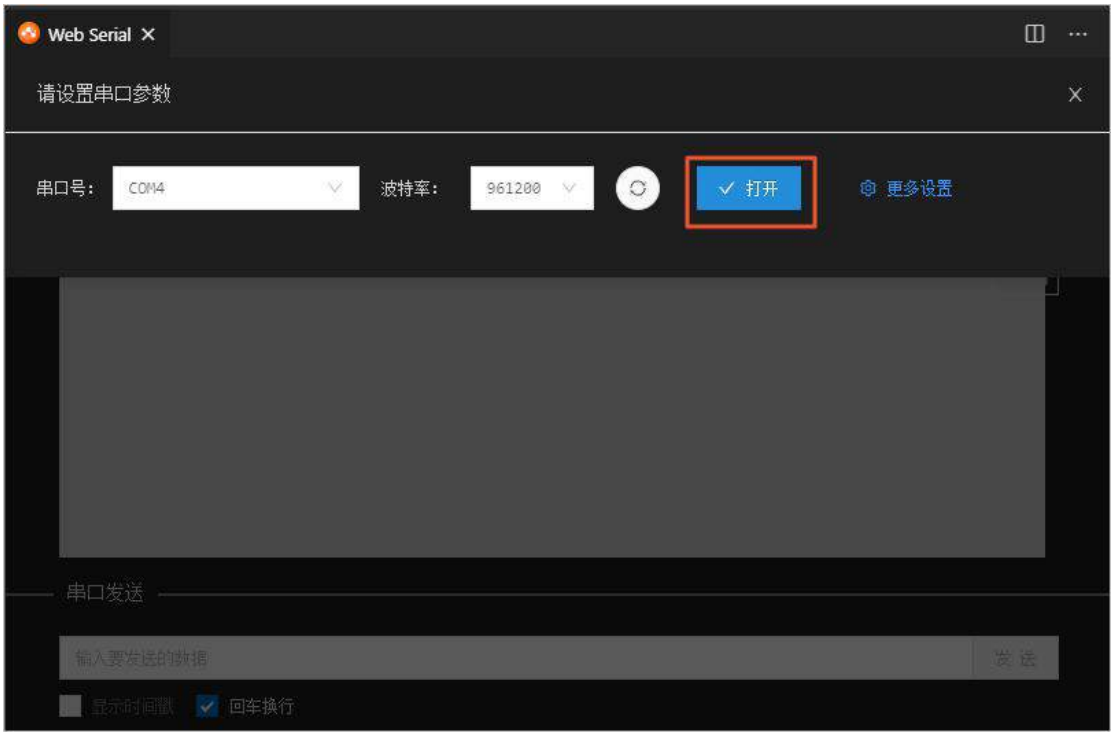
a. **在弹出的 Web Serial 窗口，单击串口号下拉框，选择 COM4。**



b. **单击波特率下拉框，选择 961200。**



3. **单击打开**。



Web Serial 控制台不断输出联网信息，说明设备运行正常。

使用线上的开发板做开发调试

当您手上没有现成可用的开发板时，也可以使用线上的开发板来调试验证您的程序。本文主要介绍线上开发板的使用流程。

一、前提条件

在使用线上开发板进行开发调试之前，您需要完成以下准备工作。

- 准备一个阿里云主账号，参见注册账号。
- 完成固件的编译工作，参见**上一节项目编译**。

二、使用限制

使用线上开发板有以下限制：

- 不能外接额外的器件。
- 不能与设备做物理交互，例如：按键交互。
- 不能观察物理的设备状态，例如：LED 灯的闪烁情况。
- 不支持 GDB 调试。

说明：线上的开发板主要用于学习验证，仅仅能提供有限的操作（例如：串口交互、设备启停和固件的烧录擦除等）。如果您是做实际的设备或者项目开发，建议您使用真实的开发板进行操作。

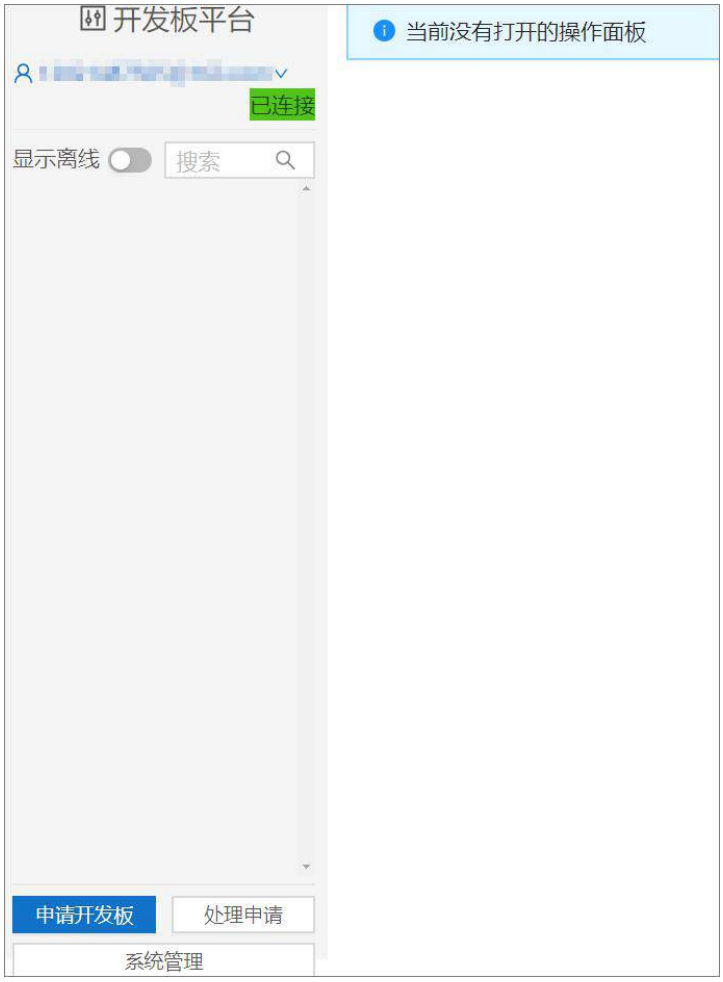
步骤一：开通服务并申请开发板

开发板平台主要为您提供线上可用的开发板，以方便大家能随时随地的学习、开发和验证。

1. 使用阿里云账号，登录开发板平台。

说明： 第一次登录时，需要阅读并同意服务协议，单击**同意并开通服务**。

2. **单击左侧下方申请开发板。**



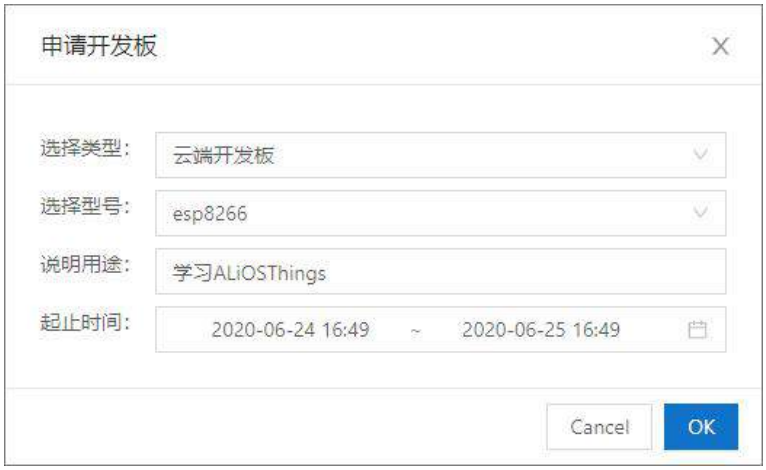
3. 在弹出的**申请开发板**对话框中配置如下信息，然后单击 **OK** 提交申请。

o **选择类型**：选择云端开发板。

o **选择型号**：选择 **esp8266**。

o **说明用途**：根据实例情况填写用途，例如：学习 ALiOSThings。

o **起止时间**：请根据实际需求选择起止时间，例如：2020-06-24 16:49~2020-06-25 16:49。

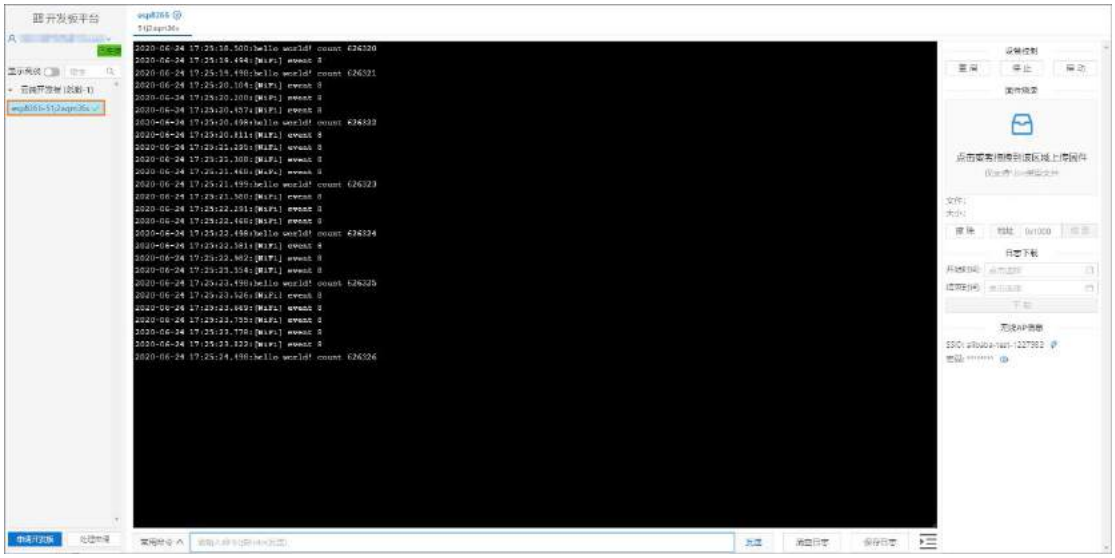


4. 等待申请审批通过以后，申请到开发板会出现在左边的设备列表中。

说明：请尽量在工作日提交申请，工作人员会在 20 分钟之内完成审批，如果在非工作日申请，审批时间可能延迟会变长。

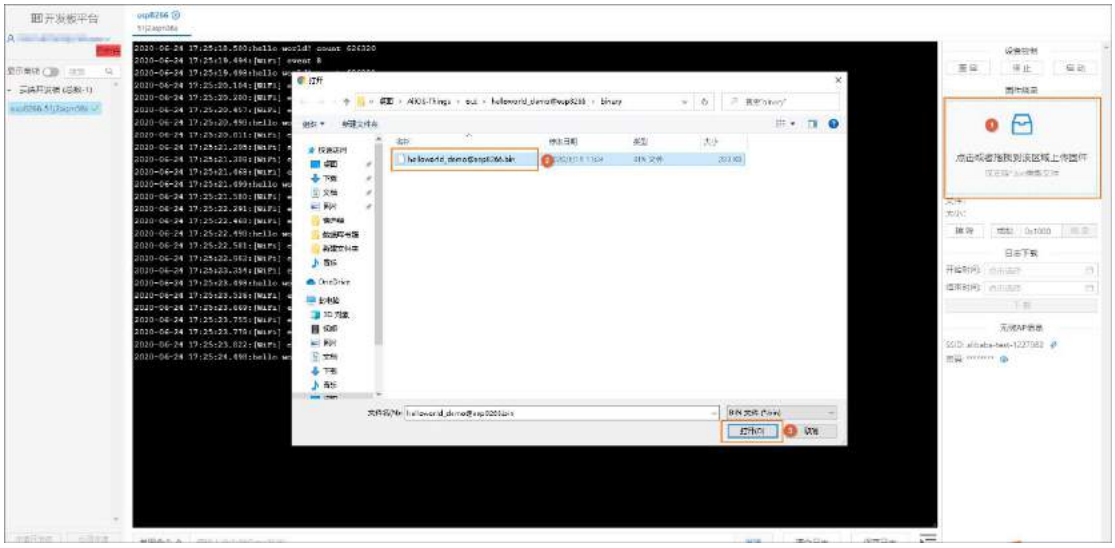
步骤二：固件烧录

1. 单击**设备名称**，打开设备操作面板。



2. 烧录您编译好的固件到设备。

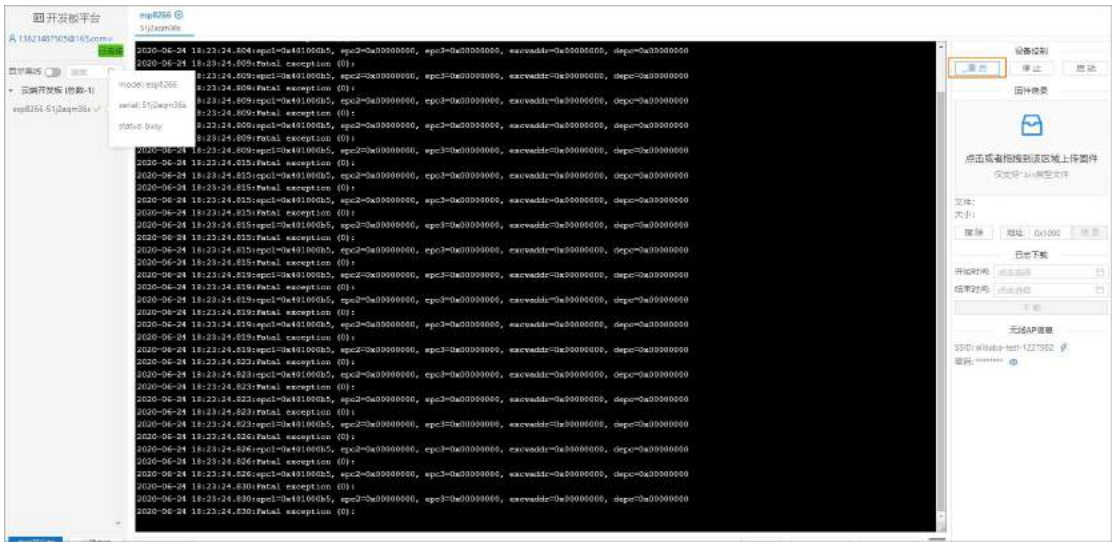
a. 单击**点击或者拖拽到该区域上传固件**，在弹出的上传界面中，选择编译好的固件，单击**打开**开始上传。



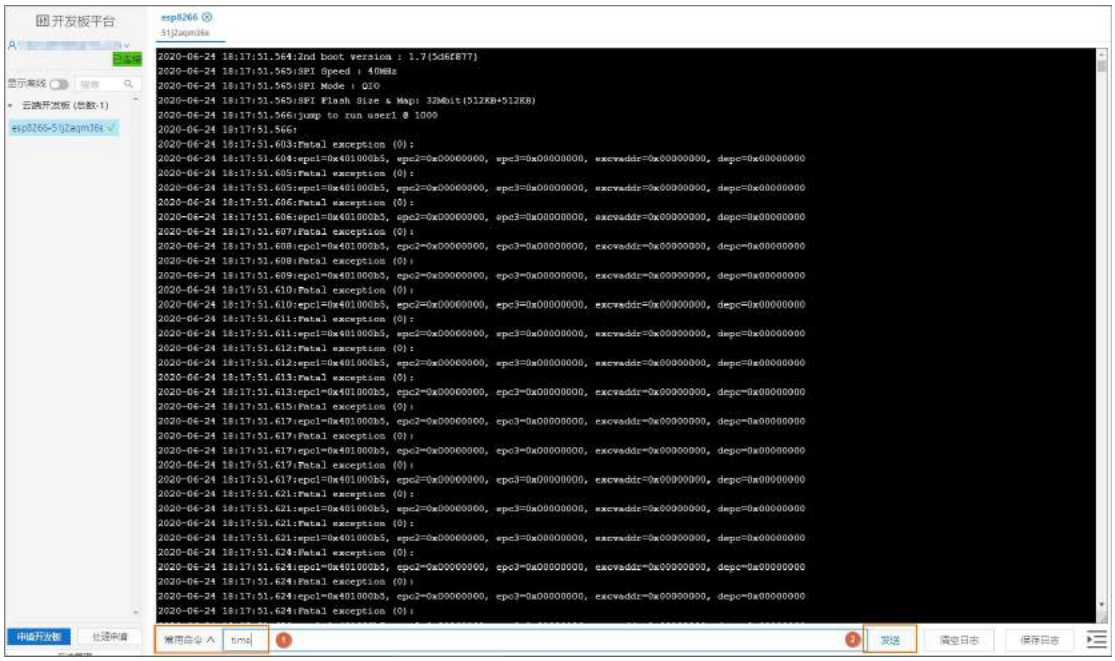
b. 上传成功后，单击**烧录**将固件刷入设备中。

步骤三：与设备交互

1. 在右侧**设备控制**区域，单击**重启**重启设备。



2. 在下方输入框，输入 time，然后单击**发送**，查看时间。



使用 AliOS Things 快速构建温度计应用

本文是基于 AliOS Things 3.1 快速构建温度计的应用场景。涉及 AliOS Things 组件开发，构建 AliOS Things 用户项目，AliOS Things HAL API 使用，向 AliOS Things 中添加并使用组件。用到的硬件设备有半导体开发板、温度传感器、数码管显示器。

一、背景信息

- **AliOS Things HAL API**

AliOS Things 向用户提供的统一硬件抽象 API，为用户提供标准的对 MCU 片上硬件操作的能力。使得用户应用在多个不同平台之间迁移时，可以无需考虑硬件平台本身的区别，同时也使外设驱动类组件可以快速地在适配过 AliOS Things 的不同硬件之间复用，提高开发效率。

- **AliOS Things 组件**

AliOS Things 提供的功能扩展能力。通过将外设驱动，通信协议，控制算法，数据算法等通用能力封装为符合 AliOS Things 标准的组件，是各种附加能力可以快速被用户复用于自己的应用中。同时，组件与 OS 分离设计，按需安装使用，可以有效地控制资源占用。

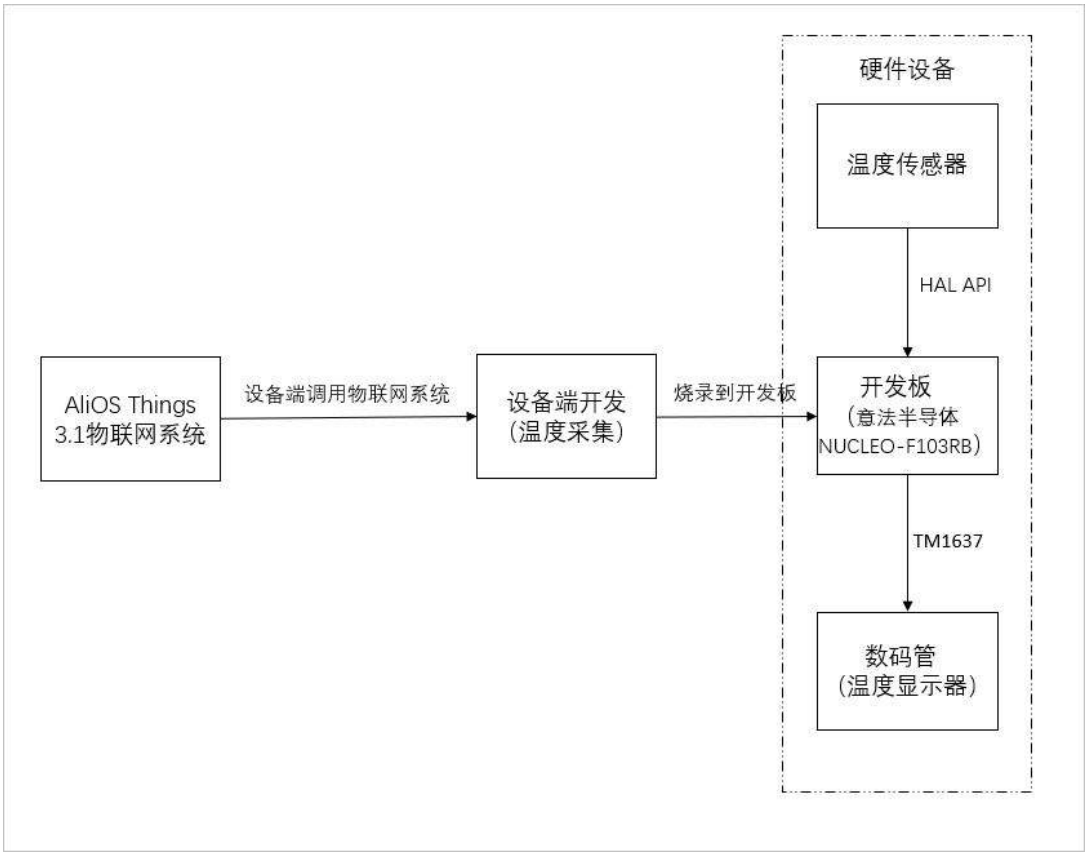
- **I2C 总线**

I2C 总线用于总线上的器件之间传送信息，是一种简单、双向二线制同步串行总线。发送数据一方称为主器件，被寻址的器件称为从器件，由主器件负责产生定时时钟和终止数据传送。

- **TM1637 芯片**

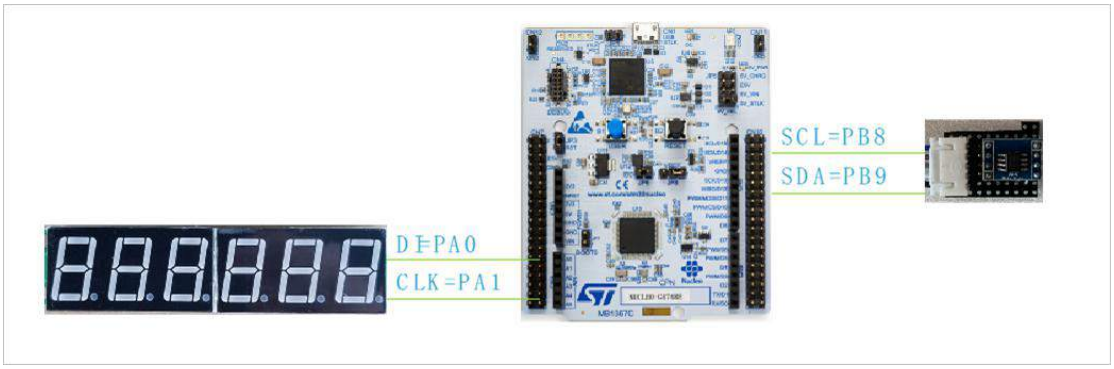
TM1637 是一种带键盘扫描接口的 LED（发光二极管显示器）驱动控制专用电路，内部集成有 MCU 数字接口、数据锁存器、LED 高压驱动、键盘扫描等电路。主要应用于电磁炉、微波炉及小家电产品的显示屏驱动。

整体流程图如下：



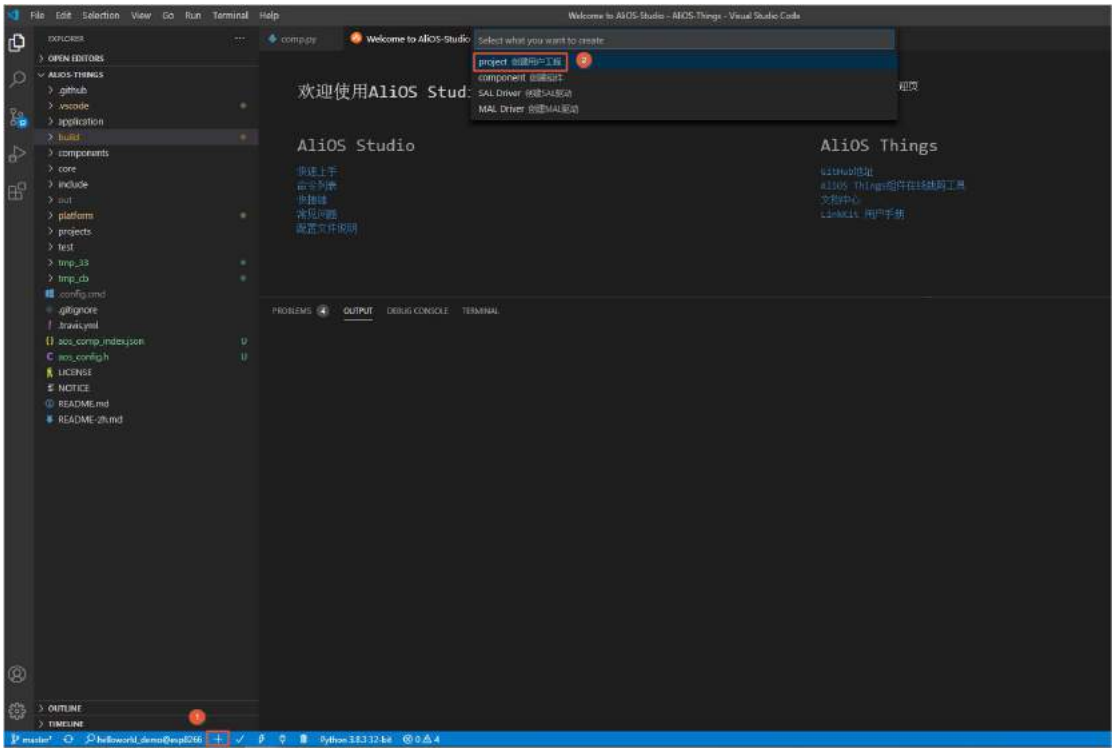
- **物联网系统**使用 AliOS Things 3.1 版本作为基础软件平台。
- **设备端开发**使用 AliOS Things 的 HAL（硬件抽象层） API 及组件，实现片上外设的使用及外部传感器的数据采集。
- **开发板**采用意法半导体 NUCLEO-F103RB 开发板，主控芯片为 stm32f103RBT6。AliOS Things 已完成对该主板的适配。
- **温度传感器**采用 LM75A 数字温度传感器，该传感器通过 I2C 总线与主控 MCU 通信，可以提供 0.125℃精度的温度输出。
- **数码管**采用由 TM1637 驱动的 6 位数码管进行显示。

硬件连接图如下：

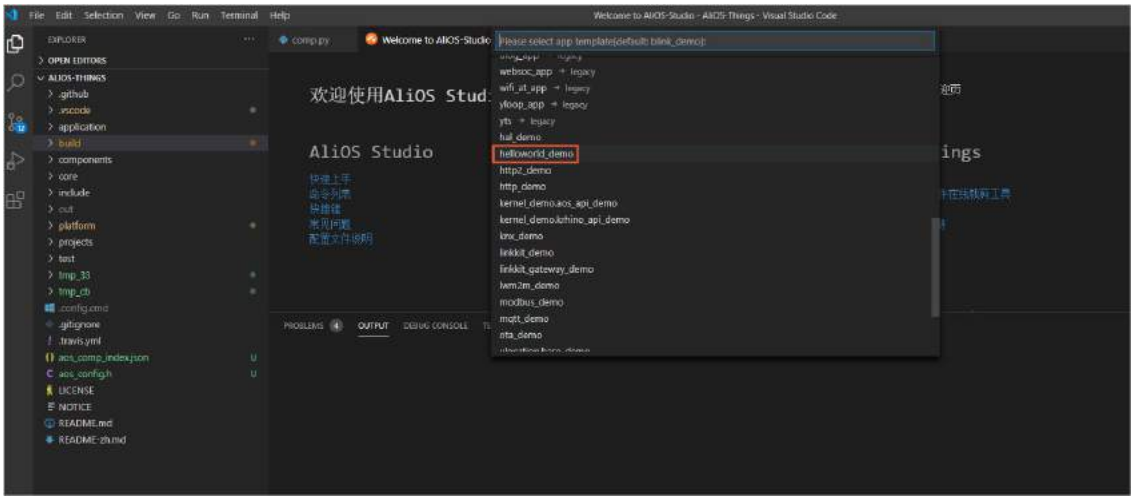


步骤一：创建 AliOS Things 用户项目

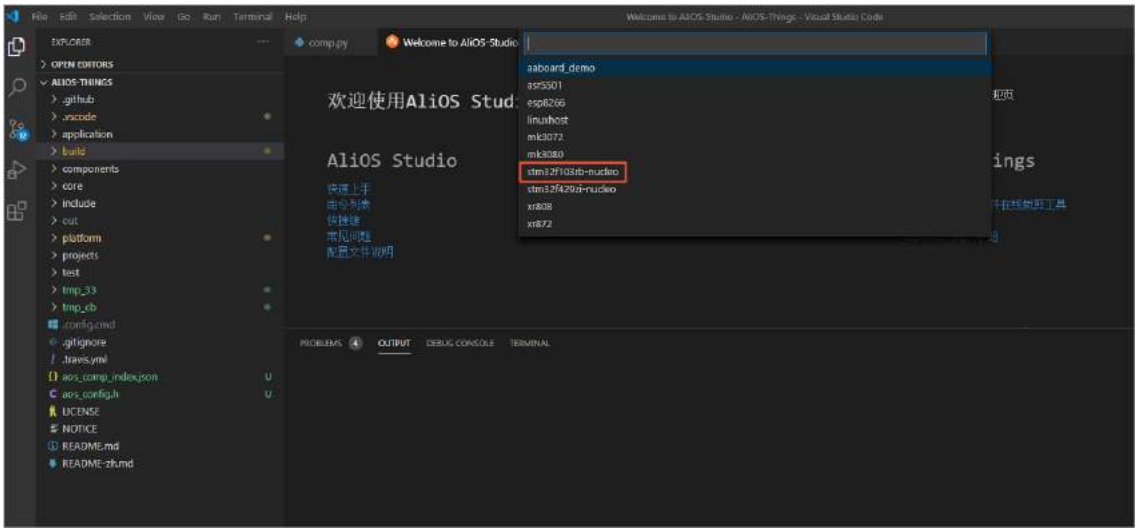
1. 单击左下角+按钮。在弹窗的选项框中选择 **project**。



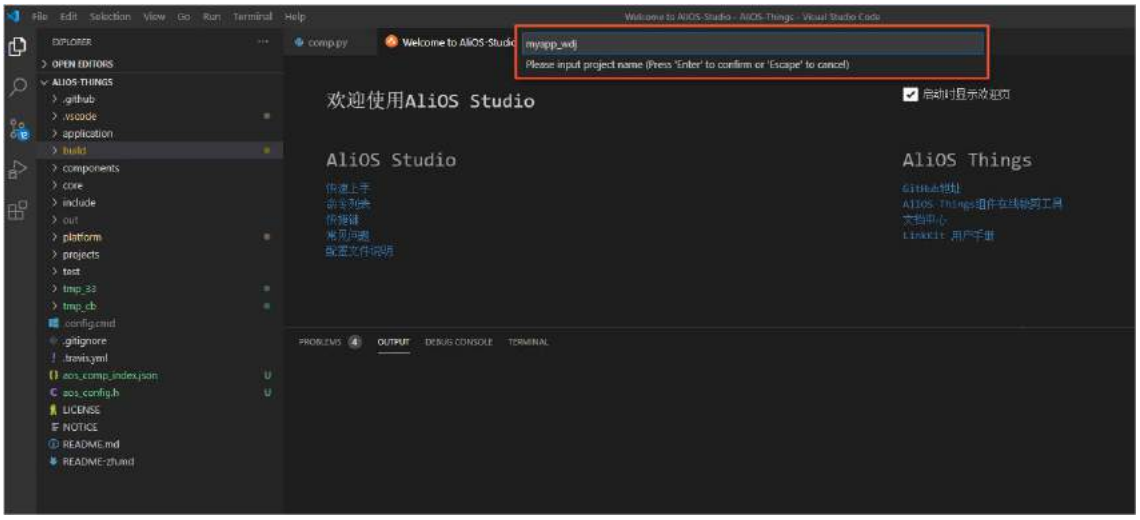
2. 选择 **helloworld_demo** 作为工程构建模板。



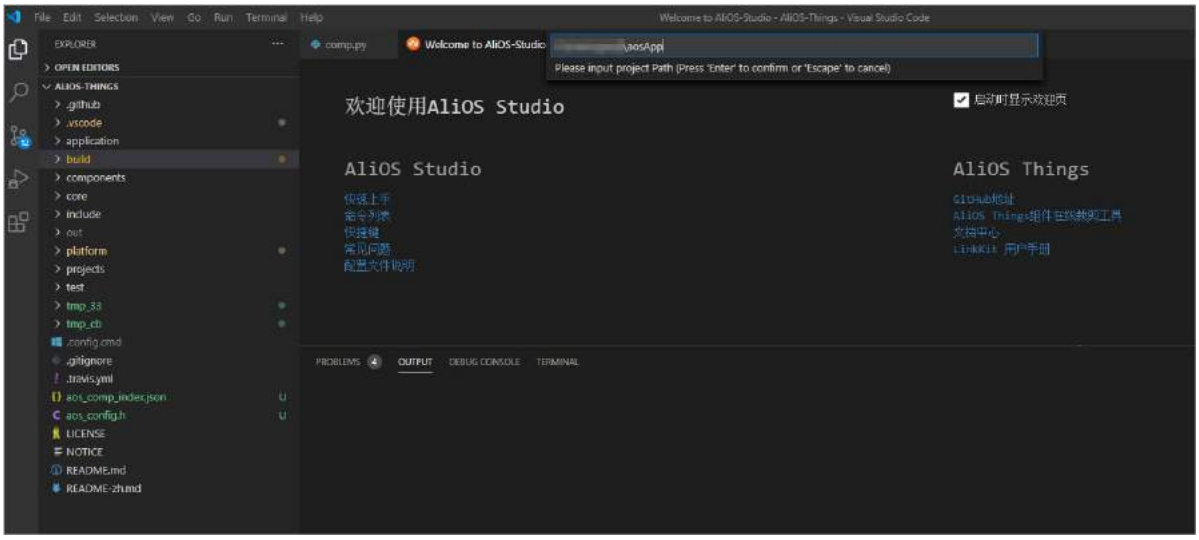
3. 选择 **stm32f103rb-nucleo** 作为开发板。



4. 输入应用名，按 **enter** 键，例如：myapp_wdj。

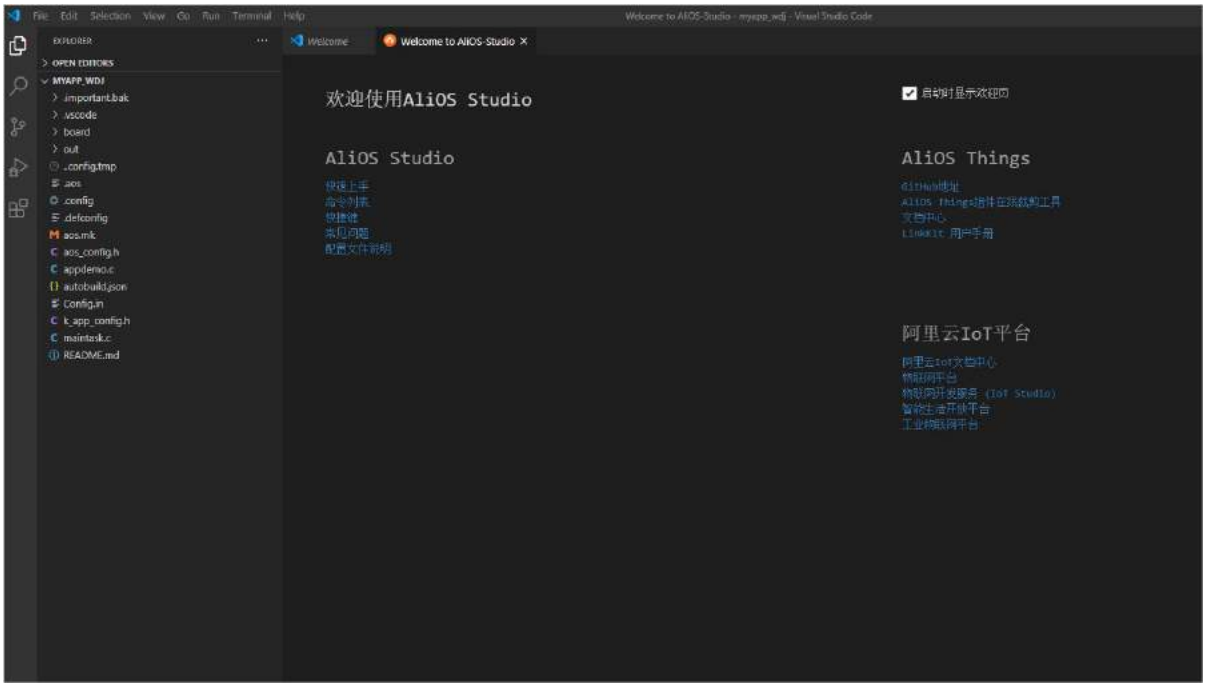


5. 选择工程存储目录，按 **enter** 键生成项目工程。



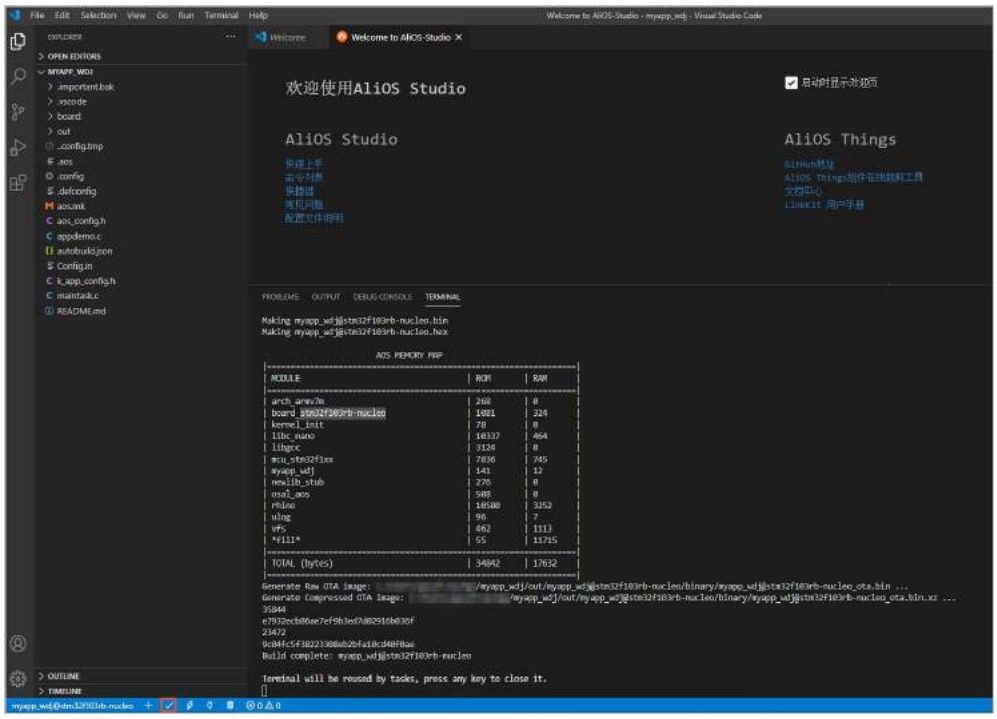
6. 开发工具 Visual Studio Code 自动打开生成的用户应用。

用户应用主入口位于 appdemo.c 文件中。



7. 单击下面的编译图标  进行编译。

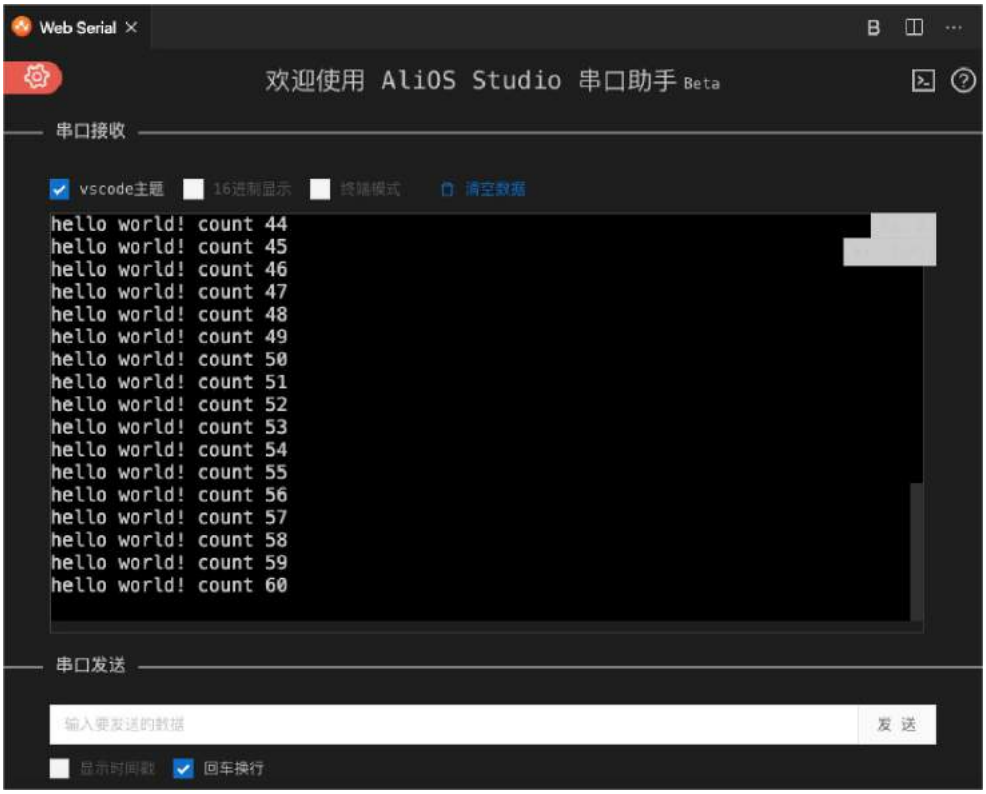
在控制台中看到编译结果。



8. 单击下面的烧录  图标，将编译好的固件烧写到开发板中。

9. 单击**插头**图标，可以打开串口调试工具进行观察。

开发板已在按照模板程序打印 hello world 字符及打印计数。



至此，用户应用工程建立完毕。

步骤二：加入温度采集

1. 在工程 appdemo.c 文件中加入如下代码。

```
#include <aos/hal/i2c.h>              // 包含需要用到的 I2C HAL 驱动
#define I2C1_PORT_NUM PORT_I2C_1      // 使用开发板的第一路 I2C 通道
#define I2C_RX_TIMEOUT 10             // 定义接收超时时间宏
i2c_dev_t i2c1;                       // 定义一个 I2C 设备结构体，用户后续操作具体的 I2C 接口
/* i2c 初始化函数 */
void i2c1_init(void)
{
    int ret = -1;
    /* i2c attr config */
    i2c1.port = PORT_I2C_1;
```

```
    i2c1.config.address_width = I2C_HAL_ADDRESS_WIDTH_7BIT;
    i2c1.config.freq = I2C_BUS_BIT_RATES_400K;
    i2c1.config.mode = I2C_MODE_MASTER;
    /* init i2c1 with the given settings */
    ret = hal_i2c_init(&i2c1);
}
```

HAL 的其他参数细节请参考文档。

2. 在方法 application_start 中加入 I2C 初始化。

```
int application_start(int argc, char *argv[])
{
    int count = 0;
    printf("nano entry here!\r\n");
    //fd = board_lcd_create("name");
       //board_lcd_write(fd,buffer,len);
    i2c1_init();                    // 调用 I2C 初始化
  while(1) {
       printf("hello world! count %d \r\n", count++);
       aos_msleep(1000);
    };
}
```

3. 加入温度采集及相关数据处理部分到主循环中。

```
int application_start(int argc, char *argv[])
{
    int count = 0;
    int ret = -1;
    float temp;
    unsigned char
    i2c_data_buf[10];
```

```
    uint16_t TempAll;
    int sig = 1;
    int temp_int;
    printf("nano entry here!\r\n");
    i2c1_init();                    // 调用 I2C 初始化
    while(1) {
        // printf("hello world! count %d \r\n", count++);
        /*通过 HAL 读取  0x90 地址设备的 0x00 寄存器，读取长度为 2 字节，返回数据放入 i2c_data_buf
中*/
        ret = hal_i2c_mem_read(&i2c1, 0x90, 0x00,
I2C_MEMADD_SIZE_8BIT,i2c_data_buf,2,I2C_RX_TIMEOUT);
        if(ret != 0)
        {
            printf("read fail!\r\n");
            // return;
    }
        TempAll = ((i2c_data_buf[0] << 8) + i2c_data_buf[1]);    // 根据 LM75 自身特性拼接温度数据
        printf("middle=%d\r\n",TempAll);
        if ((i2c_data_buf[0] & 0x80) != 0)                    // 判断数据的符号
        {
            TempAll = ~(TempAll) + 1;
            sig = -1;
        }
        TempAll >>=5;      // 低 5 位无效数据除
        printf("middle=%d\r\n",TempAll);    // 打印中间数据用于调试
        temp = TempAll * 0.125* sig;       // 计算最终数据
        printf("temp = %.2f\n",temp);       // 打印最终数据
        aos_msleep(1000);
```

```
};

}
```

4. 再次编译、烧录、串口监控。

在串口调试框中看到数据已完成采集。至此，温度采集部分完成。

步骤三：加入本地显示

1. 组件包安装。

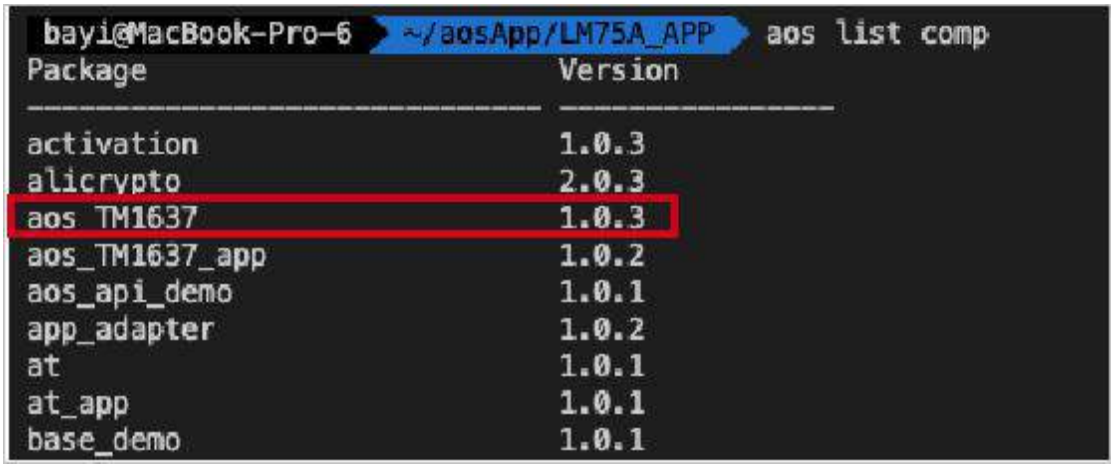
在组件包所在目录，输入如下安装命令。

```
aos install comp -L aos_TM1637-1.0.3.zip
```

aos-cube 工具会把组件安装到 AliOS Things 源码目录。

输入以下命令查看安装的组件。

```
aos list comp
```



2. 在应用中加入组件。

a. 在应用中进行组件的头文件引用。

加入的代码如下：

```
#include <aos_TM1637.h>                    // 包含组件头文件
```

b. 在应用中加入组件提供的相关 API，完成显示设备驱动输出。

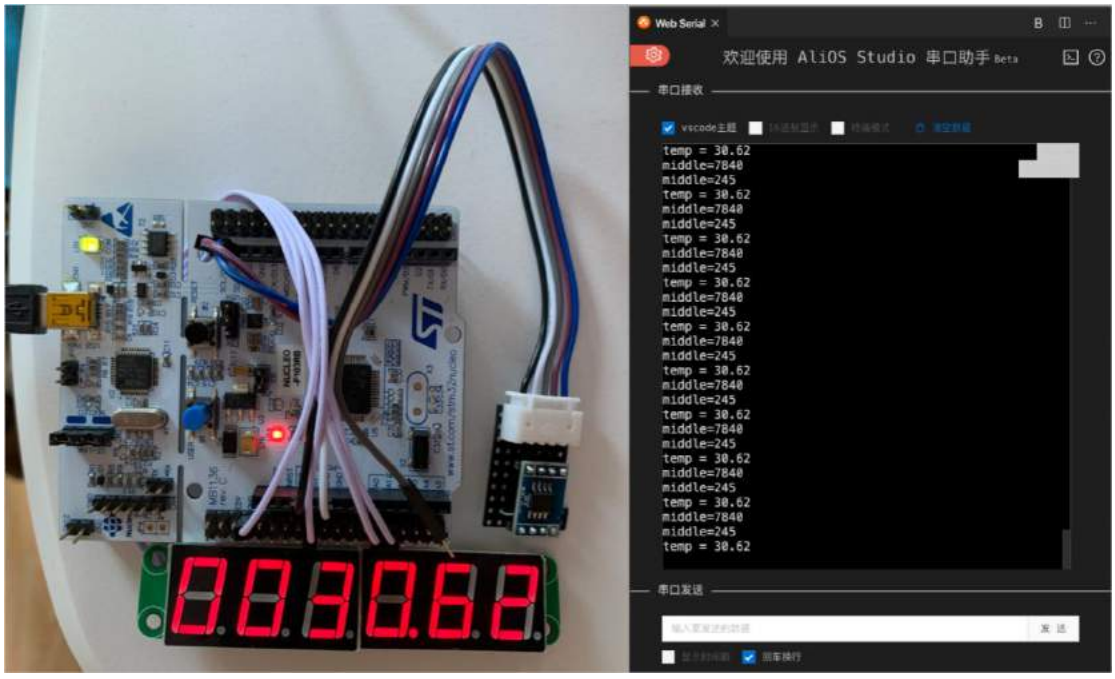
加入的代码如下：

```
aos_drv_TM1637_init(1,0);          // 组件初始化，PA1=clk；PA0=data
```

```
aos_drv_TM1637_SetBrightness(3);      // 通过组件设置数码管亮度
temp_int = temp * 100;                // 数码管不支持小数显示，先将温度转为整数
aos_drv_TM1637_DisplayDecimal(temp_int,2);  // 通过组件 API 使数码管显示，并将小数点定到第二位以匹配真实数据
```

3. 再次编译、烧录、进行串口监控。

看到温度数据显示到了数码管上，和串口输出一致。



appdemo.c 文件的完整代码如下：

```
/*
 * Copyright (C) 2015-2020 Alibaba Group Holding Limited
 */
#include <stdio.h>
#include <stdlib.h>
#include <aos/kernel.h>
#include "aos/init.h"
#include "board.h"
#include <k_api.h>
```

```
#include <aos/hal/i2c.h>                    // 包含需要用到的 I2C HAL 驱动
#include <aos_TM1637.h>                     // 包含组件头文件
#define I2C1_PORT_NUM    PORT_I2C_1         // 使用开发板的第一路 I2C 通道
#define I2C_RX_TIMEOUT 10                   // 定义接收超时时间宏
i2c_dev_t i2c1;                             // 定义一个 I2C 设备结构体，用户后续操作具体的 I2C 接口
/* i2c 初始化函数 */
void i2c1_init(void)
{
    int ret = -1;
    /* i2c attr config */
    i2c1.port                = PORT_I2C_1;
    i2c1.config.address_width = I2C_HAL_ADDRESS_WIDTH_7BIT;
    i2c1.config.freq         = I2C_BUS_BIT_RATES_400K;
    i2c1.config.mode         = I2C_MODE_MASTER;
    /* init i2c1 with the given settings */
    ret = hal_i2c_init(&i2c1);
}
int application_start(int argc, char *argv[])
{
    int count = 0;
    int ret = -1;
    float temp;
    unsigned char i2c_data_buf[10];
    uint16_t TempAll;
    int sig = 1;
    int temp_int;
    printf("nano entry here!\r\n");
    i2c1_init();                            // 调用 I2C 初始化
```

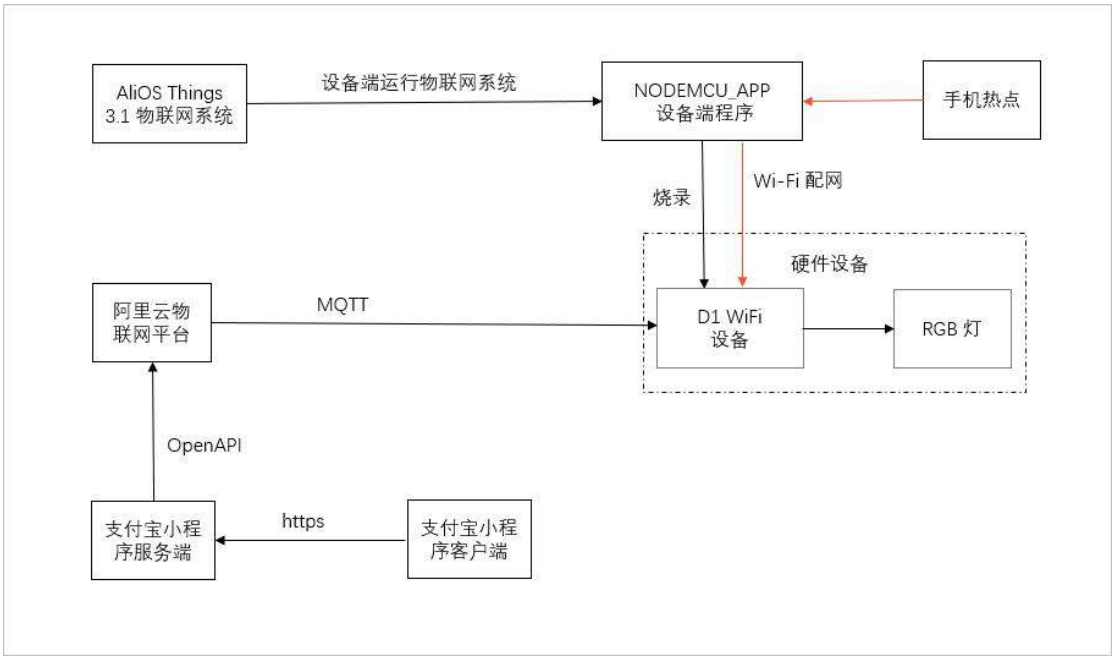
```
    aos_drv_TM1637_init(1,0);            // 组件初始化，PA1=clk；PA0=data
    aos_drv_TM1637_SetBrightness(3);     // 通过组件设置数码管亮度
    while(1) {                           // printf("hello world! count %d \r\n", count++);
        /*通过 HAL 读取 0x90 地址设备的 0x00 寄存器，读取长度为 2 字节，返回数据放入
i2c_data_buf 中*/
        ret = hal_i2c_mem_read(&i2c1, 0x90, 0x00, I2C_MEMADD_SIZE_8BIT,i2c_data_buf,2,
I2C_RX_TIMEOUT);
        if(ret != 0)
        {
            printf("read fail!\r\n");          // return;
        }
        TempAll = ((i2c_data_buf[0] << 8) + i2c_data_buf[1]);    // 根据 LM75 自身特性拼接温度数据
        printf("middle=%d\r\n",TempAll);
        if ((i2c_data_buf[0] & 0x80) != 0)                       // 判断数据的符号
        {
            TempAll = ~(TempAll) + 1;
            sig = -1;
        }
        TempAll >>= 5;          // 低 5 位无效数据除
        printf("middle=%d\r\n",TempAll);      // 打印中间数据用于调试
        temp = TempAll * 0.125 * sig;         // 计算最终数据
        printf("temp = %.2f\n",temp);         // 打印最终数据
        temp_int = temp * 100;                // 数码管不支持小数显示，先将温度转为整数
        aos_drv_TM1637_DisplayDecimal(temp_int,2);  // 通过组件 API 使数码管显示，并将小数点
定到第二位以匹配真实数据
        aos_msleep(1000);
    };
}
```

使用 AliOS Things 快速构建 RGB 灯应用

本文将基于 AliOS Things 3.1 系统搭建应用，此应用通过支付宝小程序控制 RGB 灯的颜色。

一、背景信息

整体流程图如下：



- **支付宝小程序服务端**主要功能是为支付宝小程序提供 API SaaS 服务，同时通过 OpenAPI SDK 对接阿里云物联网（IoT）平台。
- **支付宝小程序客户端**主要功能是显示前端控制界面，通过 https API 发送指令控制设备属性。
- **设备端程序**主要功能是基于 AliOS Things 3.1 系统适配 D1 WiFi 设备开发。
- **阿里云物联网平台**主要功能是提供安全可靠的设备连接通信能力，支持设备数据采集上云，规则引擎流转数据和云端数据下发设备端。此外，也提供方便快捷的设备管理能力，支持物模型定义，数据结构化存储，和远程调试、监控、运维。
- **硬件设备**主要功能是通过 D1 WiFi 设备连接网络，接收支付宝小程序的指令控制 RGB 灯的颜色。D1 WiFi 设备的芯片型号是 ESP 8266。

步骤一：管理阿里云物联网平台设备

支付宝小程序和设备端是通过阿里云 IoT 平台进行通信的，具体是通过设备四元组信息进行连接的。所以要先在阿里云物联网平台创建产品，在产品下添加对应型号的设备，产生该设备四元组信息。要有阿里云账号并开通阿里云物联网平台服务。

1. 登录阿里云物联网平台。
2. 创建产品。
   a. 单击左侧菜单栏的**设备管理>产品**，打开产品列表页。

   b. 单击**创建产品**进入创建产品页面。
   c. 参考说明配置产品信息，然后单击**保存**。

- **产品名称：**输入产品名称，例如：xw 智能灯。
- **所属品类：**此处选择标准品类下的智能生活/电工照明/灯。

3. 单击**请选择标准品类**选择框。

4. 在搜索框里输入灯，然后单击搜索图标。

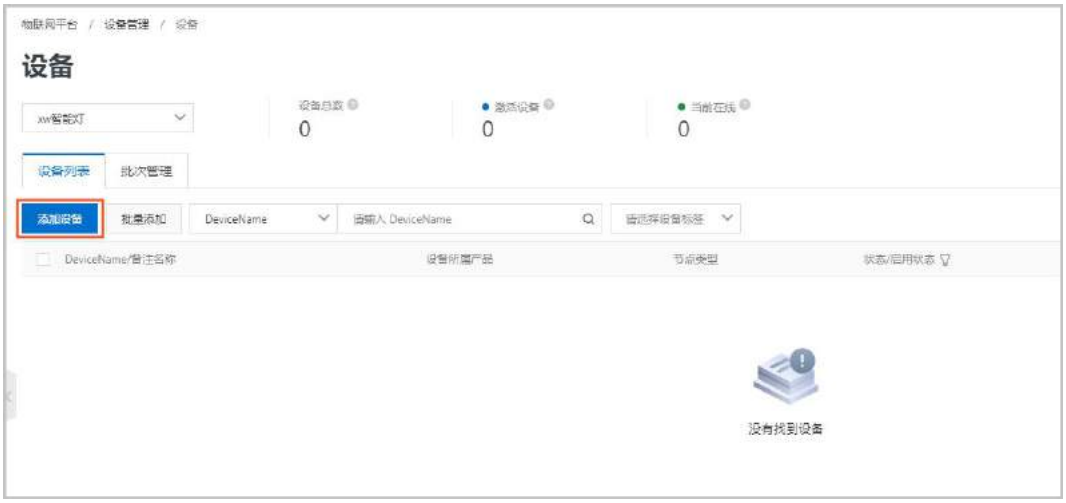
5. 单击下图所示**选择**，选择品类。

- **联网方式：**此处选择 Wi-Fi。
- **数据格式：**此处选择 ICA 标准数据格式（Alink JSON）。

6. 添加设备。

a. 单击图示中**前往添加**进入产品所属设备列表页面。

b. **单击添加设备。**

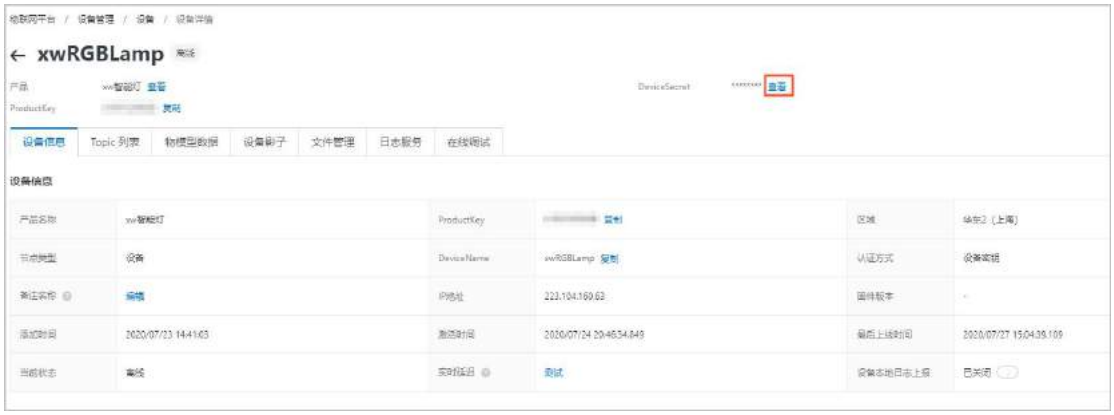


c. **参考说明配置设备信息，然后单击确认。**

- **DeviceName**：输入 DeviceName，例如：xwRGBLamp。
- **备注名称**：输入备注名称，例如：xw 智能灯设备。

d. 单击下图所示**前往查看**。

e. 单击 DeviceSecret 旁边的**查看**。



f. 单击**一键复制**复制设备三元组。

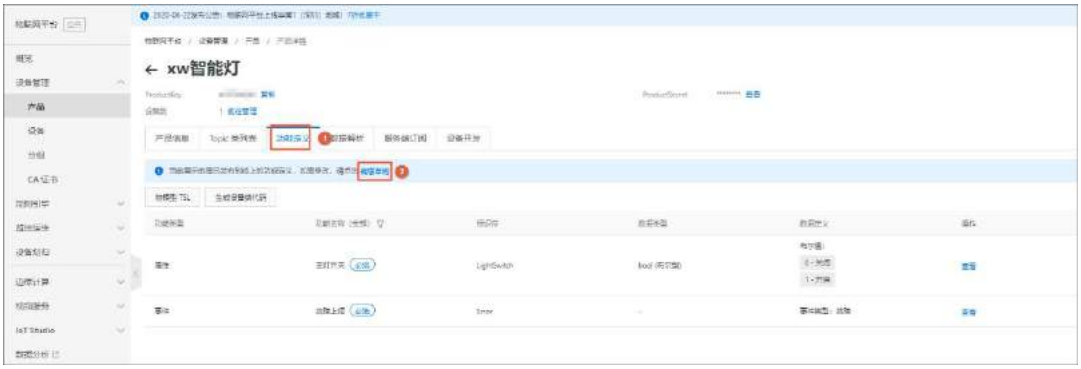
设备三元组信息如下：

```
{
  "ProductKey": "a1****kJ",
  "DeviceName": "xwRGBLamp",
  "DeviceSecret": "be8f*********b45297fc"
}
```

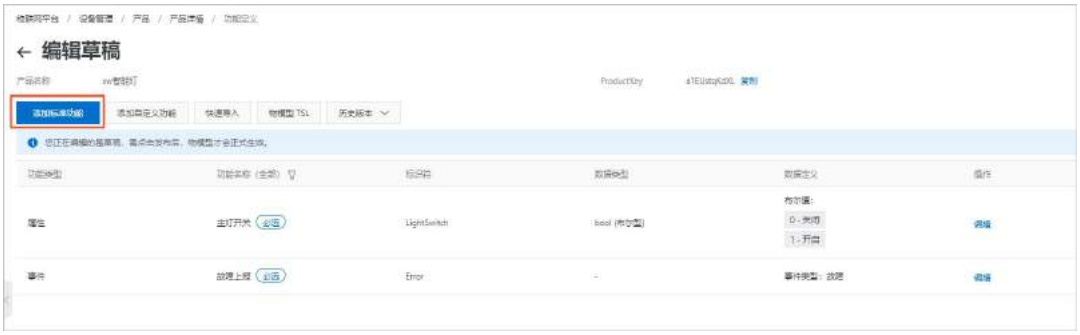
7. 添加功能。

a. 单击左侧菜单栏**产品**进入产品列表页。然后单击**操作**列的**查看**按钮。

b. **单击功能定义，然后单击编辑草稿。**

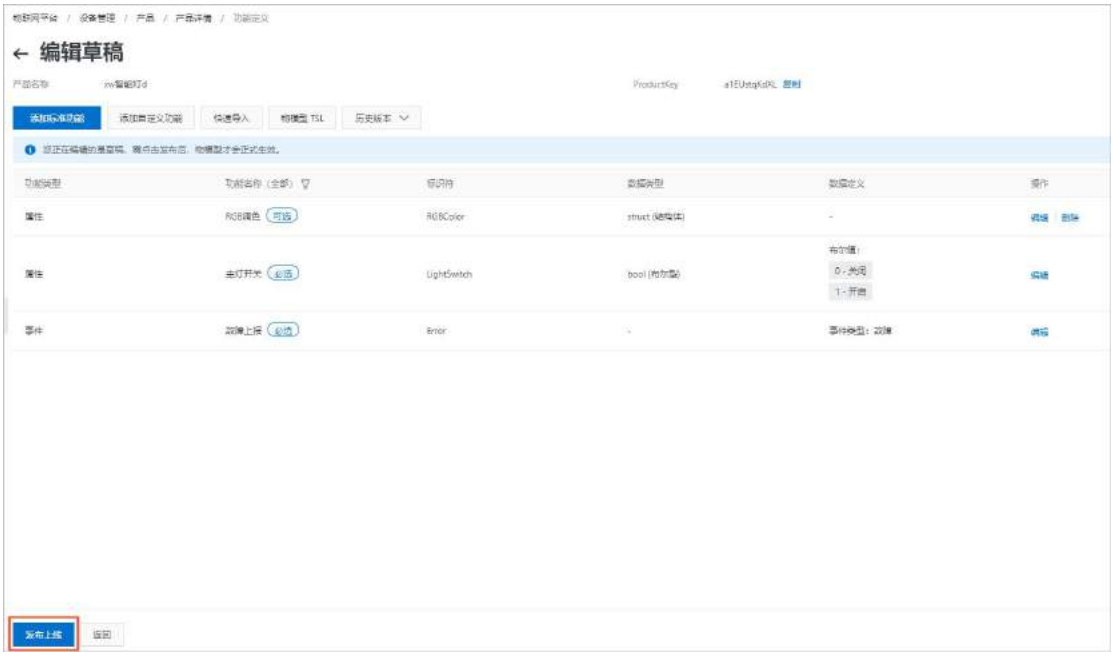


c. **单击添加标准功能。**



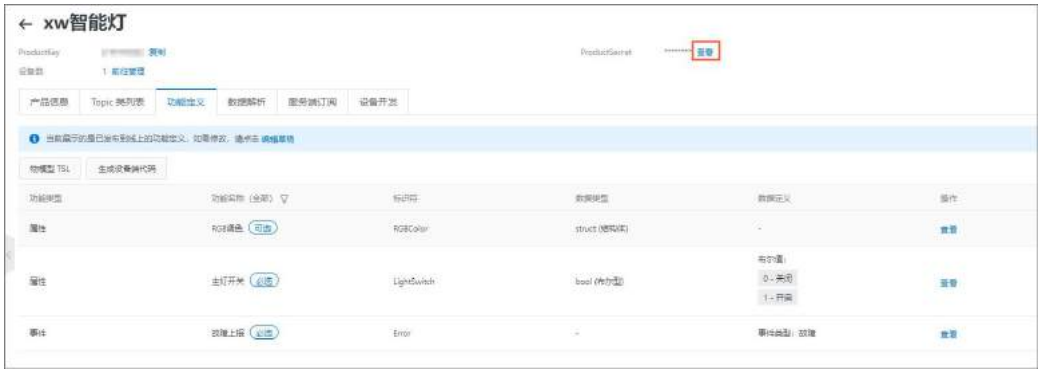
d. **选择 RGB 调色，然后单击确定。**

e. **单击发布上线。**



f. 勾选**确认已查看当前版本与线上版本的比对结果**，然后单击**确定**。

g. **单击** ProductSecret **旁边的查看。**



h. **单击** ProductSecret **栏的复制。**

ProductSecret 和上面的三元组信息组成设备的四元组信息。

设备四元组信息用于后续支付宝客户端，物联网设备端开发，此处可以保存一下。

设备四元组信息如下：

```
{
 "ProductKey": "a1******EB",
 "ProductSecret": "Sj********YX",
 "DeviceName": "xwRGBLamp",
 "DeviceSecret": "c7******************4c1c"
  }
```

8. 发布产品。

a. 在产品详情页，单击右上角的发布，弹出确认发布产品框。

b. 依次单击请确认后面的图标，然后单击发布。

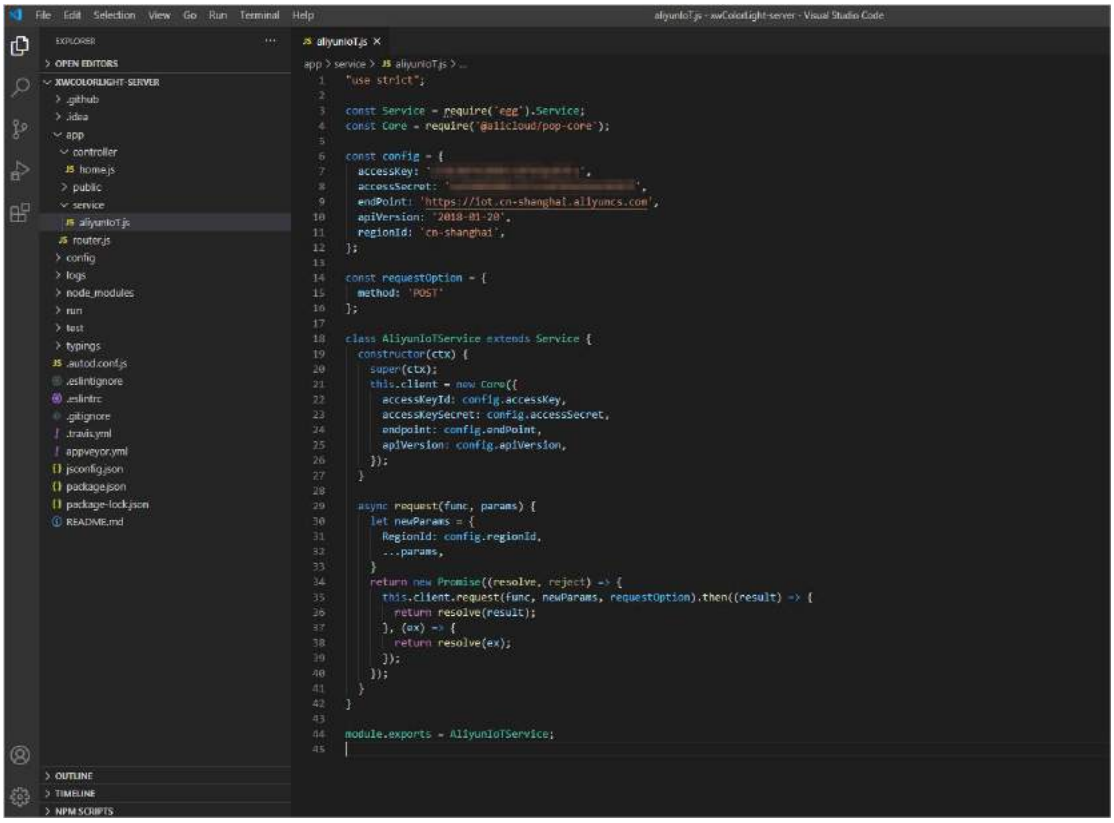
步骤二：开发支付宝小程序服务端应用

支付宝小程序服务端接收支付宝小程序客户端的指令传送到阿里云 IoT 平台。使用 Visual Studio Code 开发工具，Node.js 脚本语言。如果需要部署到线上或者上线小程序，还需要准备：

- ECS 或者公网可访问的服务器。
- 有效域名（已备案）。
- SSL 证书。

申请加入阿里巴巴小程序繁星计划，可以免费试用 ECS 和其他的小程序服务。

1. 下载安装 Node.js。
2. 下载支付宝小程序服务端源码。
3. 使用 **Visual Studio Code** 打开支付宝小程序服务端源码。



4. 修改配置信息。

修改 app/service/aliyunIoT.js 文件中的 config 属性。

```
const config = {
    accessKey: '<access-key>',
    accessSecret: '<access-secret>',
    endPoint: 'https://iot.<regionId>.aliyuncs.com',
    apiVersion: '2018-01-20',
    regionId: '<regionId>',
};
```

oaccess-key 和 access-secret 是阿里云颁发给用户访问服务所用的密钥。

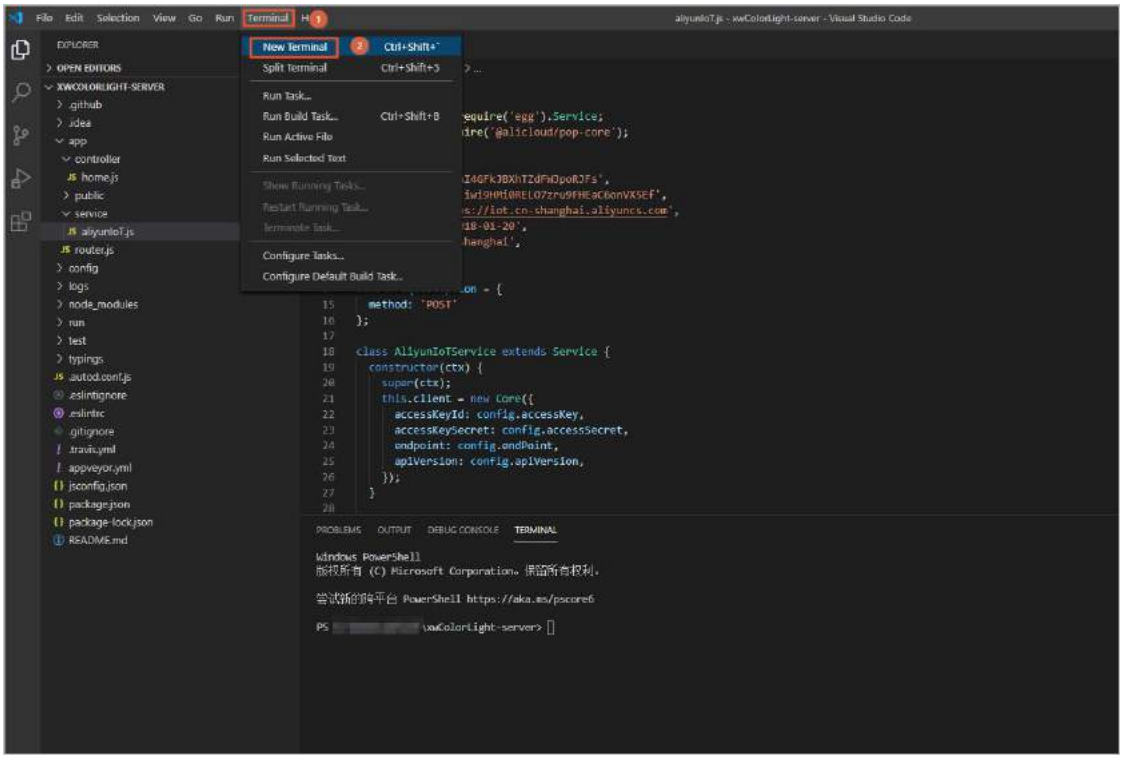
a. 登录阿里云控制台。

b. 鼠标移至右上角头像，然后单击 **AccessKey 管理**。

c. 在 AccessKey 管理页面获取 accessKey 和 accessSecret。

d. oregionId 是阿里云物联网平台所属地域，查看地域详情，例如：cn-shanghai。oendPoint 是地域所属的数据中心。阿里云物联网平台有多个地域，不同地域有不同的数据中心，例如：https://iot.cn-shanghai.aliyuncs.com。

5. 依次单击上面菜单栏的 **Terminal>New Terminal** 打开 Terminal 窗口。



6. 在 Terminal 窗口执行以下命令。

```
$ npm i

$ npm run dev
```

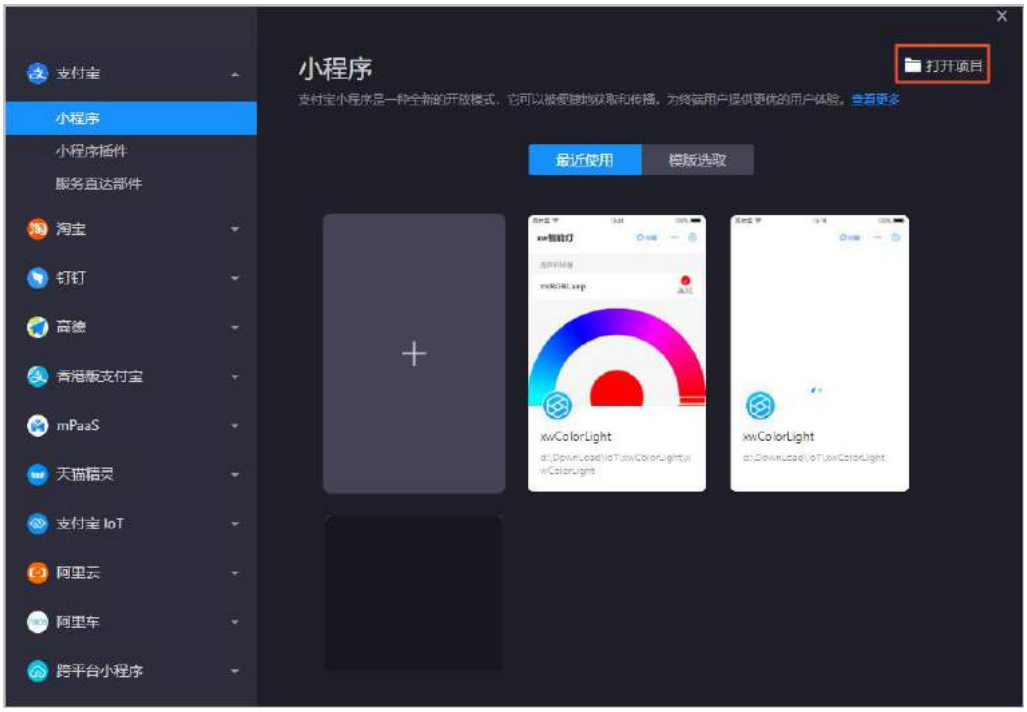
7. 使用浏览器访问 http://127.0.0.1:7001/。

说明：支付宝小程序服务端源码适用于 AliOS Things 3.1 版本。

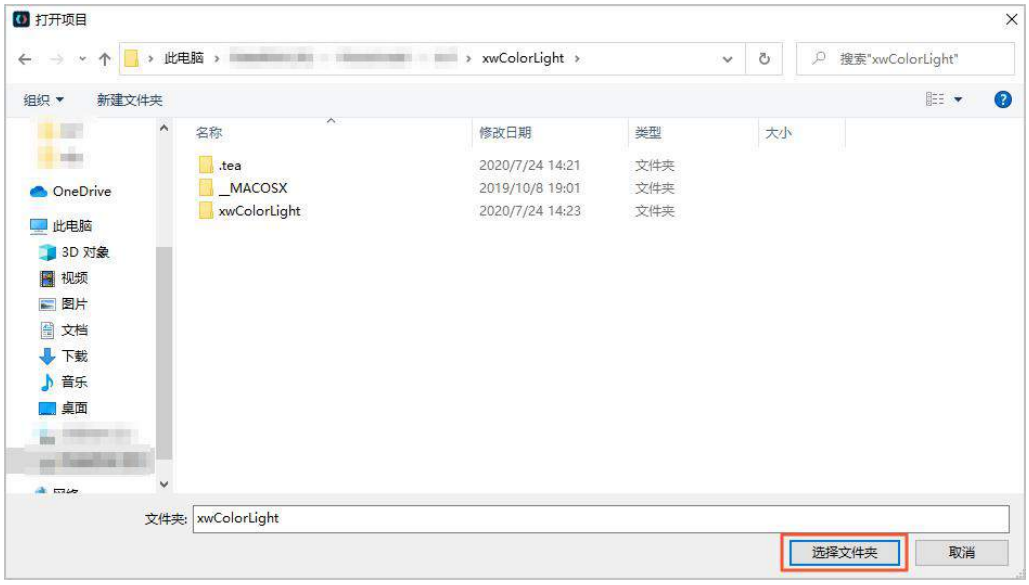
步骤三：开发支付宝小程序客户端应用

支付宝小程序客户端向支付宝小程序服务端发送 RGB 灯颜色指令。使用小程序客户端开发工具打开支付宝小程序客户端程序，并修改配置信息进行运行。了解支付宝小程序开发流程：支付宝小程序快速开始。

1. 下载安装支付宝小程序客户端开发工具。
2. 下载支付宝小程序客户端源码。
3. 用小程序开发工具打开支付宝小程序客户端源码。
    a. 打开**小程序开发者工具**客户端。然后单击客户端右上角**打开项目**。

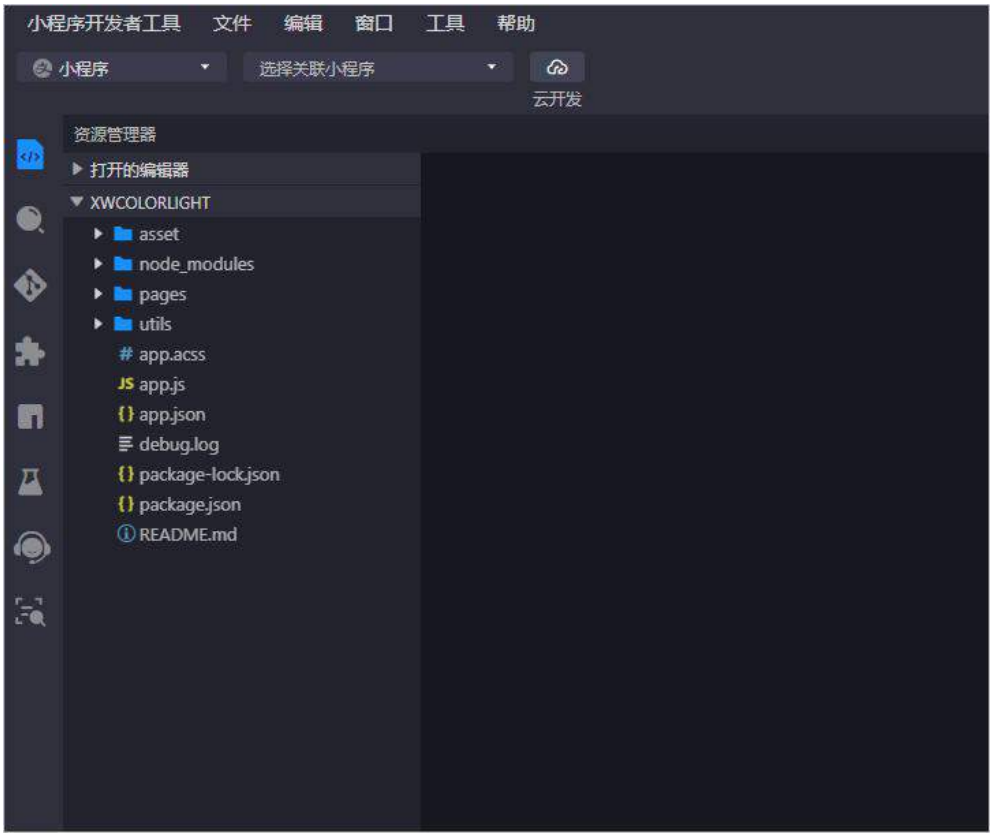


b. 选择客户端源码文件，然后单击**选择文件夹**。



c. **单击打开。**

d. 首次打开源码时提示安装依赖，单击**确认**安装依赖。



如果首次没有安装依赖，可以按下图操作手动安装所需依赖。

4. 修改配置。

打开 utils/device_api.js 文件，修改以下三个属性：

```
const defaultServerURL = '<your-domain>';

const defaultProductKey = '<product-key>';

const defaultDeviceName = '<device-name>';
```

- your-domain 为支付宝小程序服务端 API 地址，例如：http://localhost:7001/api/device。
- product-key 为设备四元组信息中 ProductKey 值。
- device-name 为设备四元组信息中 DeviceName 值。

步骤四：开发设备端应用

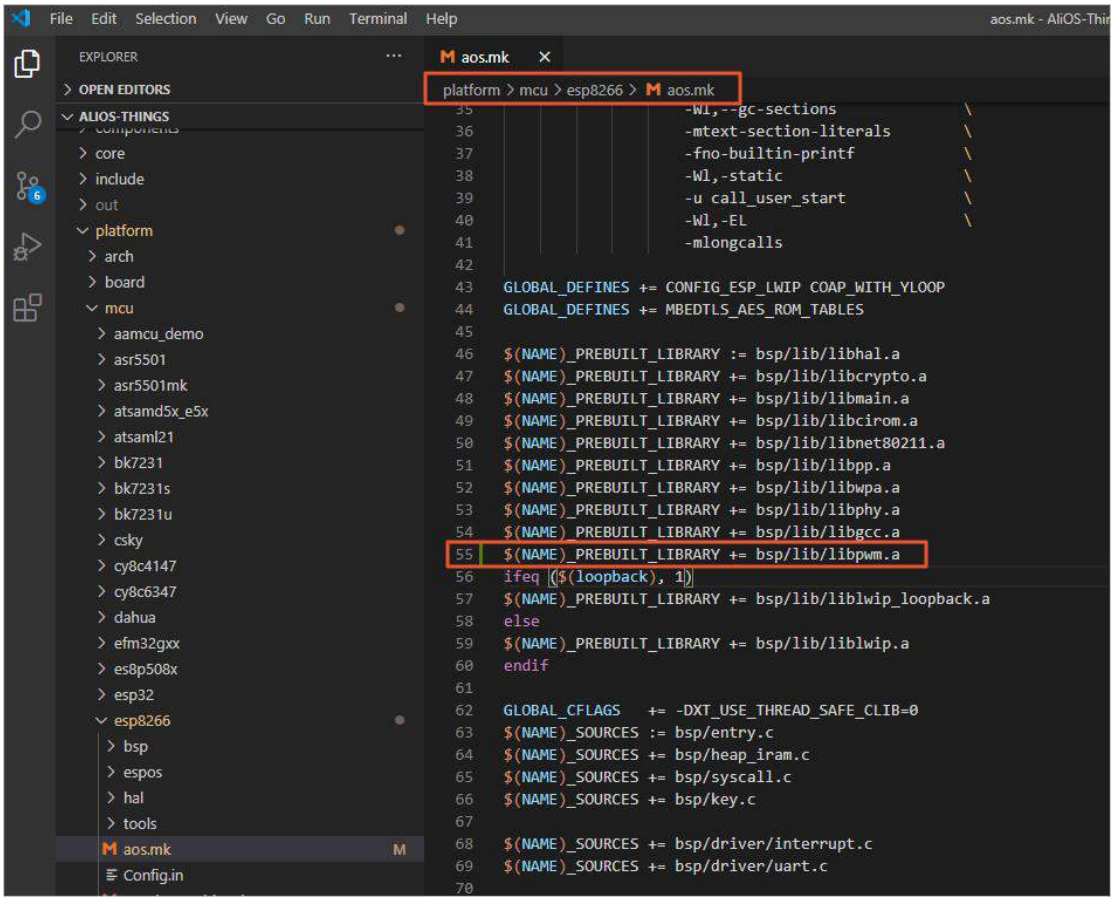
要先安装好设备对应的驱动，本例中 D1 WiFi 设备对应的驱动为 CH340。

1. 修改 AliOS Things3.1 源码。

由于设备端应用需要依赖 pwm 库，所以要在 AliOS Things 系统中加入 pwm 库。

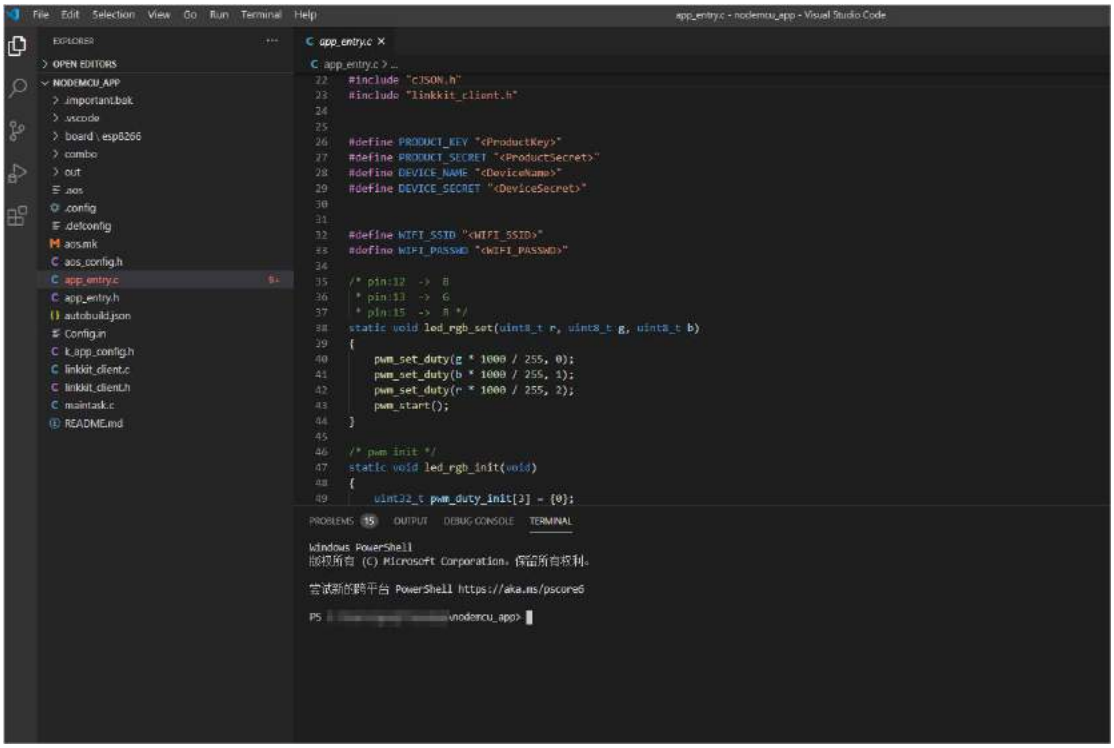
打开 platform/mcu/esp8266/aos.mk 文件，在第 54 行下面加入以下代码。

```
$(NAME)_PREBUILT_LIBRARY += bsp/lib/libpwm.a
```



2. 下载设备端源码。

3. 用开发工具 Visual Studio Code 打开设备端源码。



4. 修改设备端配置。

   a. 打开 app_entry.c 文件，修改以下属性。

设备认证信息：

```
#define PRODUCT_KEY "<ProductKey>"

#define PRODUCT_SECRET "<ProductSecret>"

#define DEVICE_NAME "<DeviceName>"

#define DEVICE_SECRET "<DeviceSecret>"
```

- ProductKey 为设备四元组信息 ProductKey 的值。
- ProductSecret 为设备四元组信息 ProductSecret 的值。
- DeviceName 为设备四元组信息 DeviceName 的值。
- DeviceSecret 为设备四元组信息 DeviceSecret 的值。

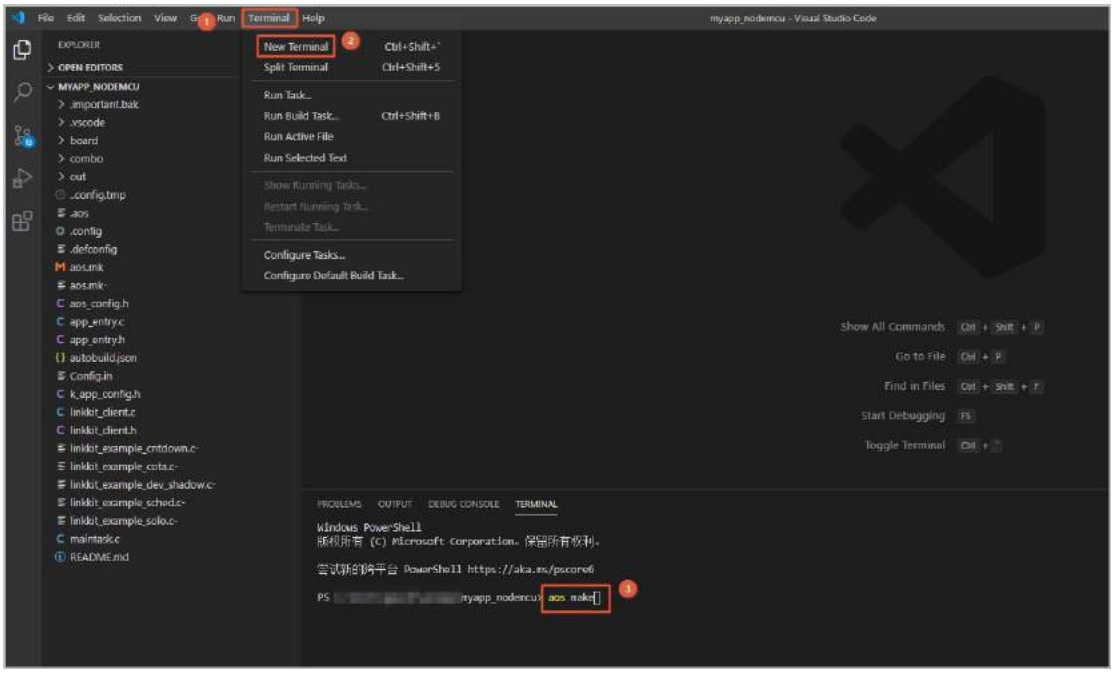
WiFi 信息：

```
#define WIFI_SSID "<WIFI_SSID>"

#define WIFI_PASSWD "<WIFI_PASSWD>"
```

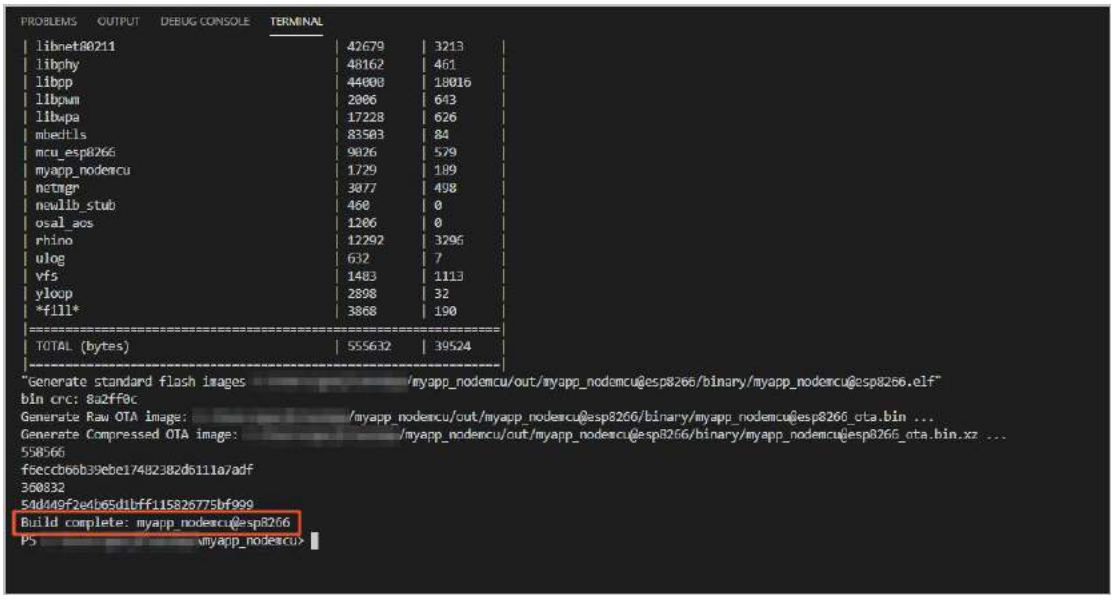
- WIFI_SSID 为手机热点的设备名称，例如：aiotesp8266。
- WIFI_PASSWD 为手机热点的密码，例如：12345678abc。

5. 编译。

单击上面导航栏的 **Terminal>New Terminal**，在下面的 Terminal 窗口输入 aos make 进行编译。



输出类似下图信息表示编译成功。

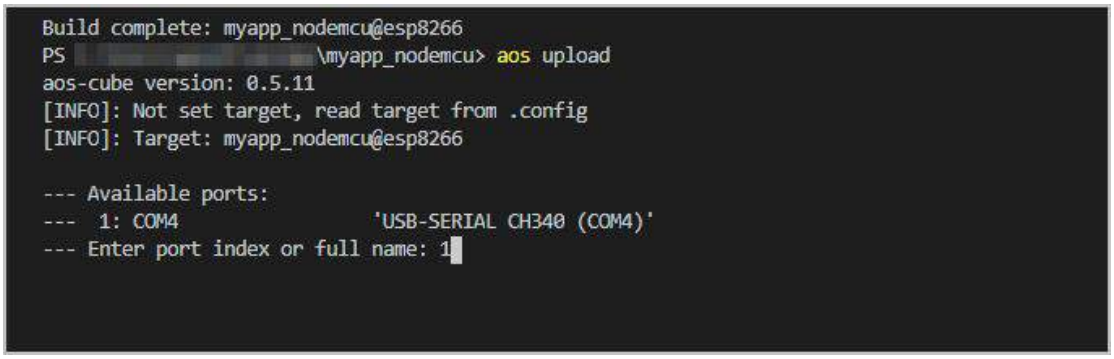


6. 烧录。

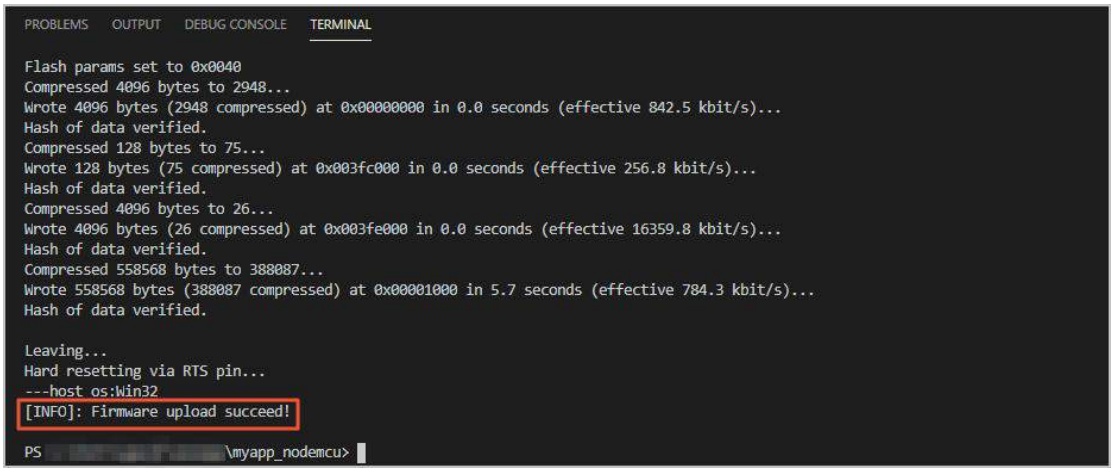
   a. 通过 USB 接口连接 D1 WiFi 设备。

   b. 在 **Terminal** 窗口输入 aos upload 进行烧录。

c. 输入 1，然后按 **enter** 键。



输出类似下图信息表示烧录成功。

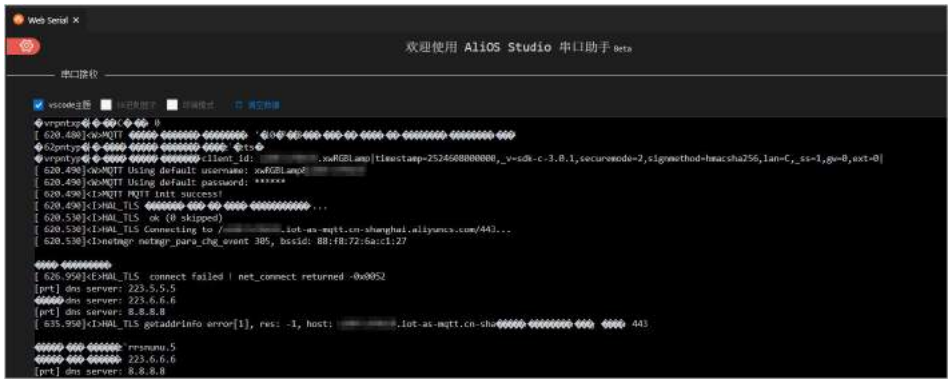


步骤五：通过手机热点进行设备配网

前面的开发工作已经完成，现在是最关键的一步。手机热点信息已经通过步骤四烧录到设备中，这里打开手机热点，设备会自动进行配网连接，通过串口监控可以查看配网日志信息。配网成功支付宝小程序客户端设备状态变为在线，阿里云 IoT 平台设备状态变为在线。

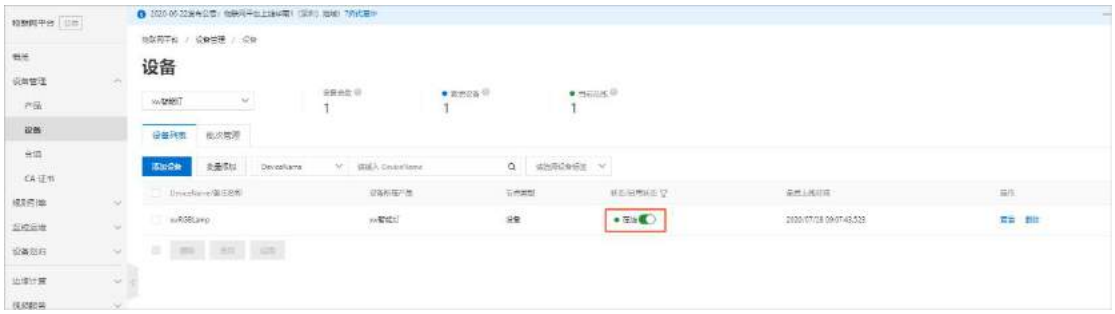
1. 打开手机热点进行网络适配。
2. 单击设备端的图标  进行串口监控，查看设备日志。
3. 选择波特率为 961200，然后单击**打开**。

Web Serial 窗口输出连接信息。



支付宝小程序客户端重新编译后，设备状态为在线。

在阿里云物联网平台查看设备状态为在线。



步骤六：通过支付宝小程序控制 RGB 灯颜色

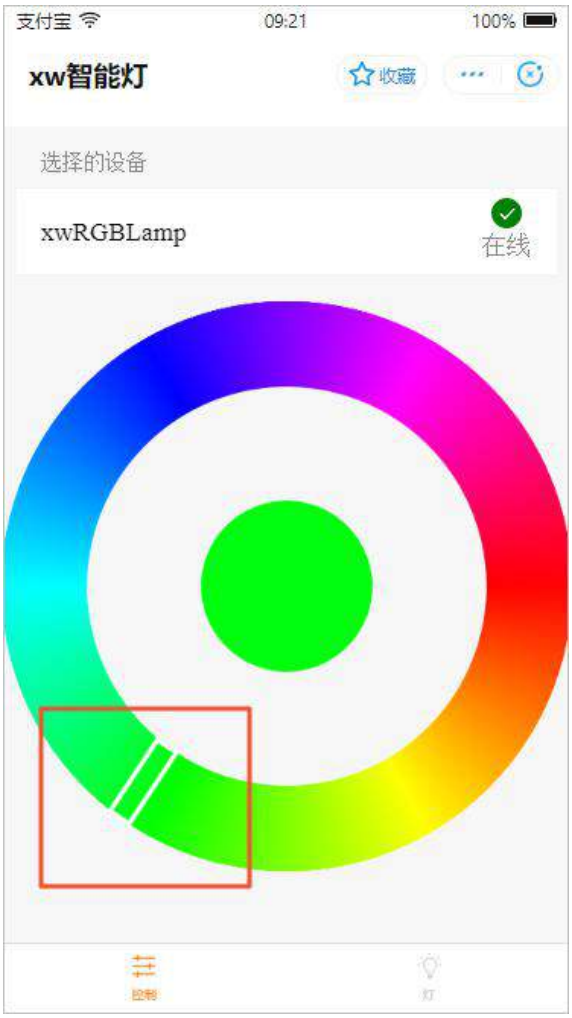
D1 WiFi 设备连上 RGB 灯，就可以通过支付宝小程序控制 RGB 灯的颜色。

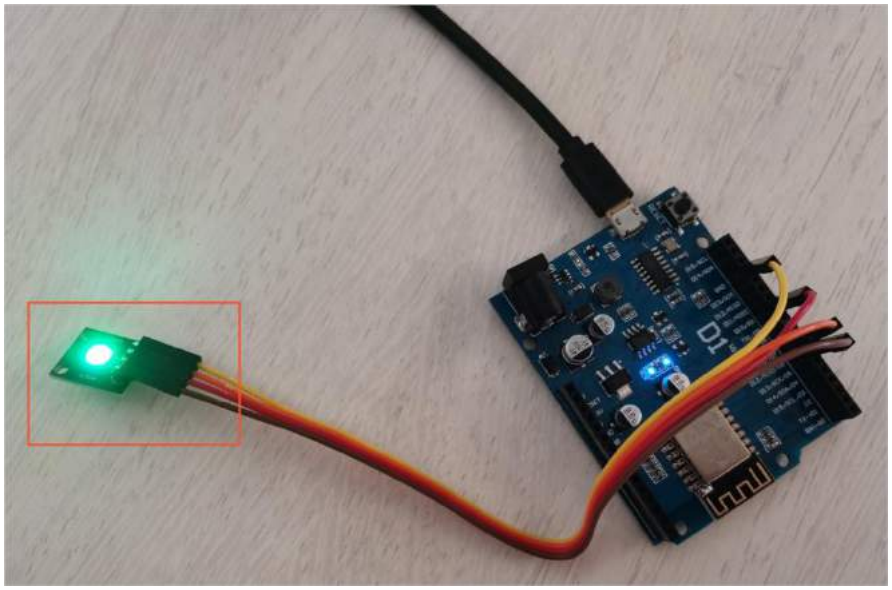
1. D1 WiFi 设备连接 RGB 灯。

接线方式如下：

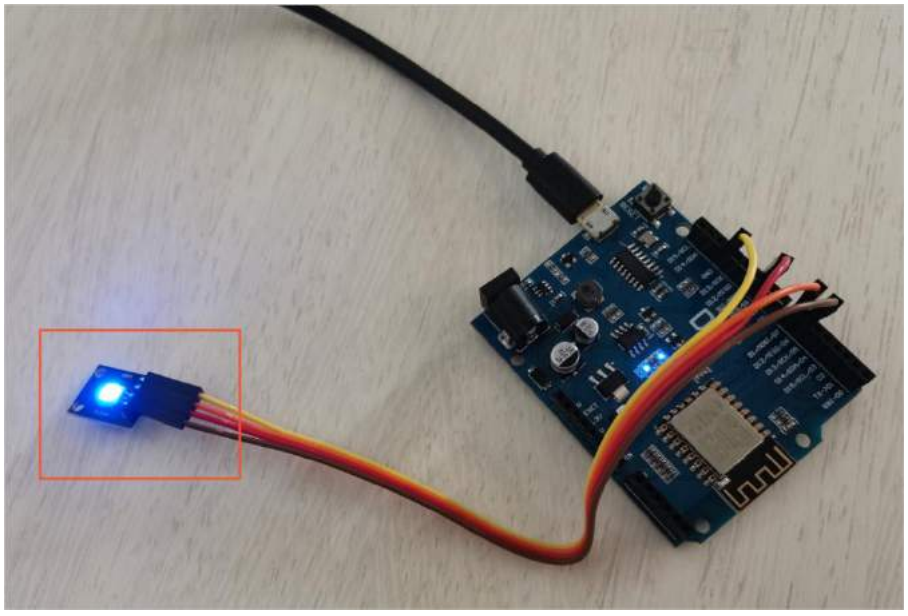
D1 WiFi 引脚	**RGB 灯引脚**
GPIO12	Blue（蓝色灯引脚）
GPIO13	Red（红色灯引脚）
GPIO15	Green（绿色灯引脚）
GND	**GND（负极）**

2. 在支付宝小程序客户端选择绿色，RGB 灯颜色变为绿色。

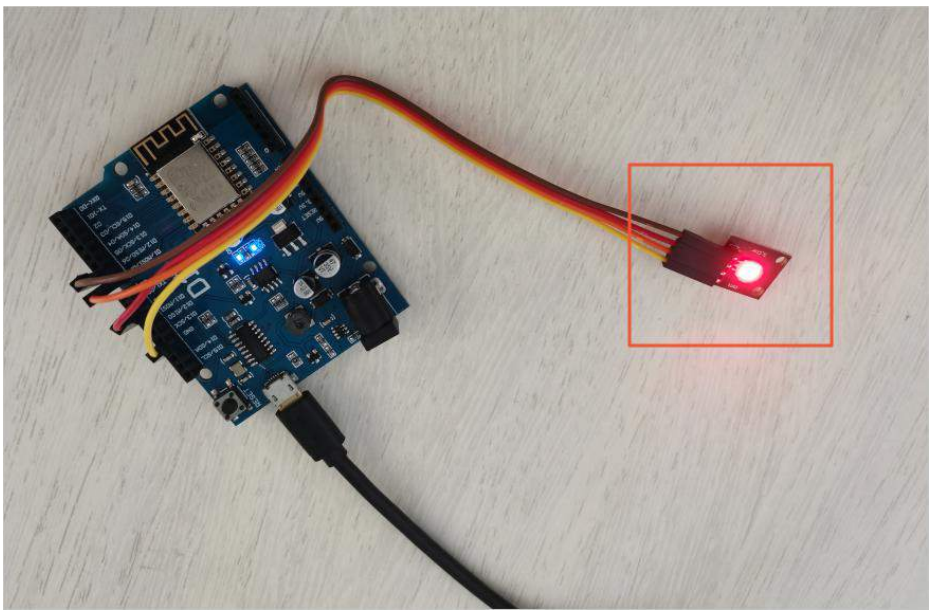

支付宝
09:21
100%
xw智能灯
收藏
选择的设备
xwRGBLamp
在线



3. 在支付宝小程序客户端选择蓝色，RGB 灯颜色变为蓝色。



4. 在支付宝小程序客户端选择红色，RGB 灯颜色变为红色。



阿里云开发者“藏经阁”

海量免费电子书下载

加入钉群

获取更多前沿资讯

加入钉群

与更多人进行技术交流