前序
1 注册GizWits账号
2 定义"微信宠物屋"产品
3 生成微信宠物屋代码
4 移植微信宠物屋驱动代码
5 编译烧录固件并进行测试
6 相关说明

前序

机智云提供了官方 **微信宠物屋** 测试固件以及相应平台的**驱动库文件**，开发者可以直接烧录并使用机智云APP进行测试测试，烧录方式请查看本文档的 **编译烧录固件并进行测试** 一节。

同时机智云也为开发者提供了SOC版与MCU版的代码自动生成功能，开发者可根据此文档进行自主开发，完成属于自己的 **微信宠物屋** 。

1 注册GizWits账号

首先登陆机智云官网，注册开发者账号。

2 定义"微信宠物屋"产品

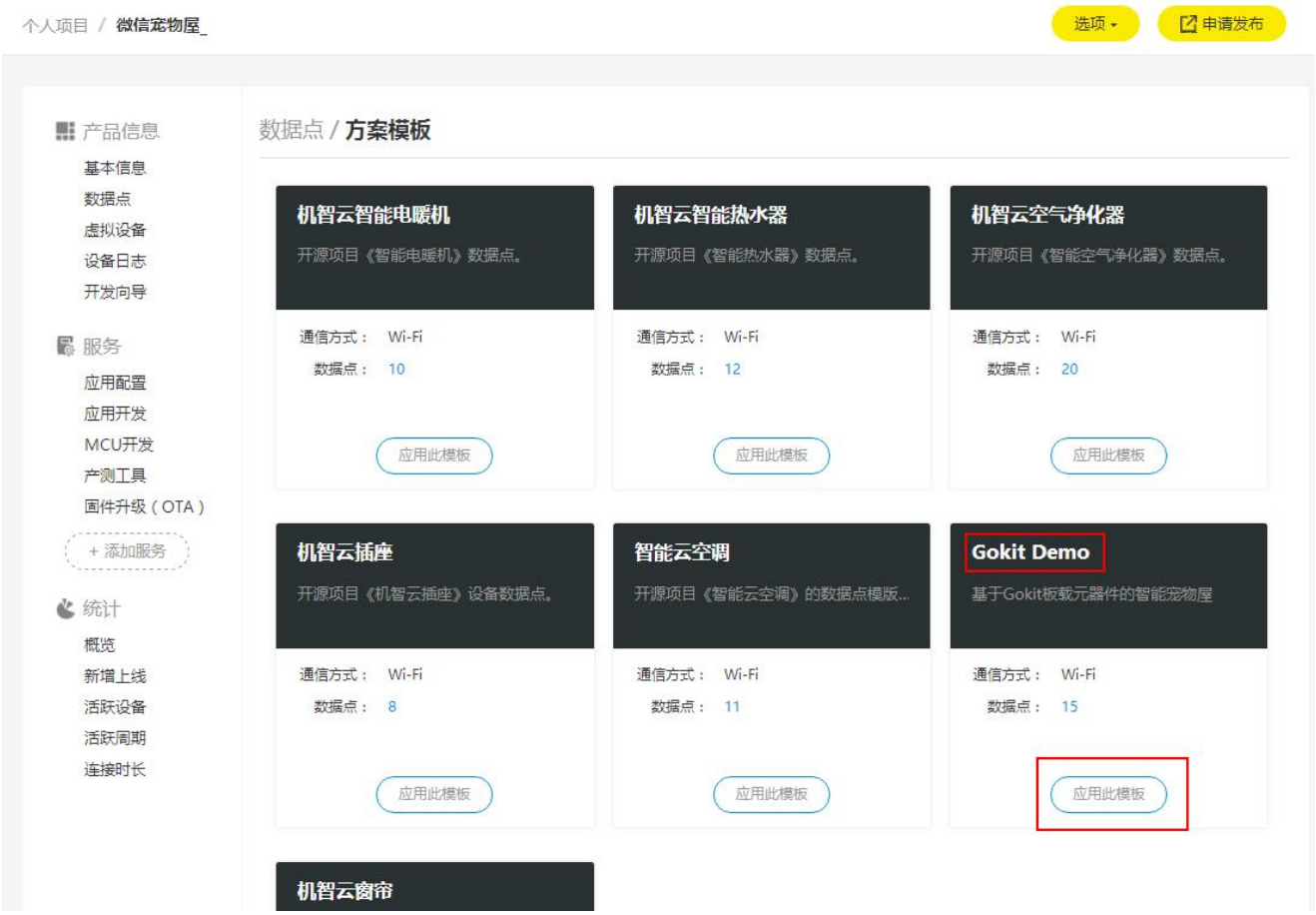
1)选择个人项目，点击创建新产品

2)输入相应的产品信息，注意这里的技术方案选择“WiFi/移动网络方案”，通信方式选择“WiFi”，最后点击“保存”

3)点击“去添加数据点”

4)在“管理”中点击“选择产品数据点模板”

5)选择“Gokit Demo”，并“应用此模板”



注：这里会导入基于Gokit板载元器件的智能宠物屋数据点模板

接下来点击“添加”

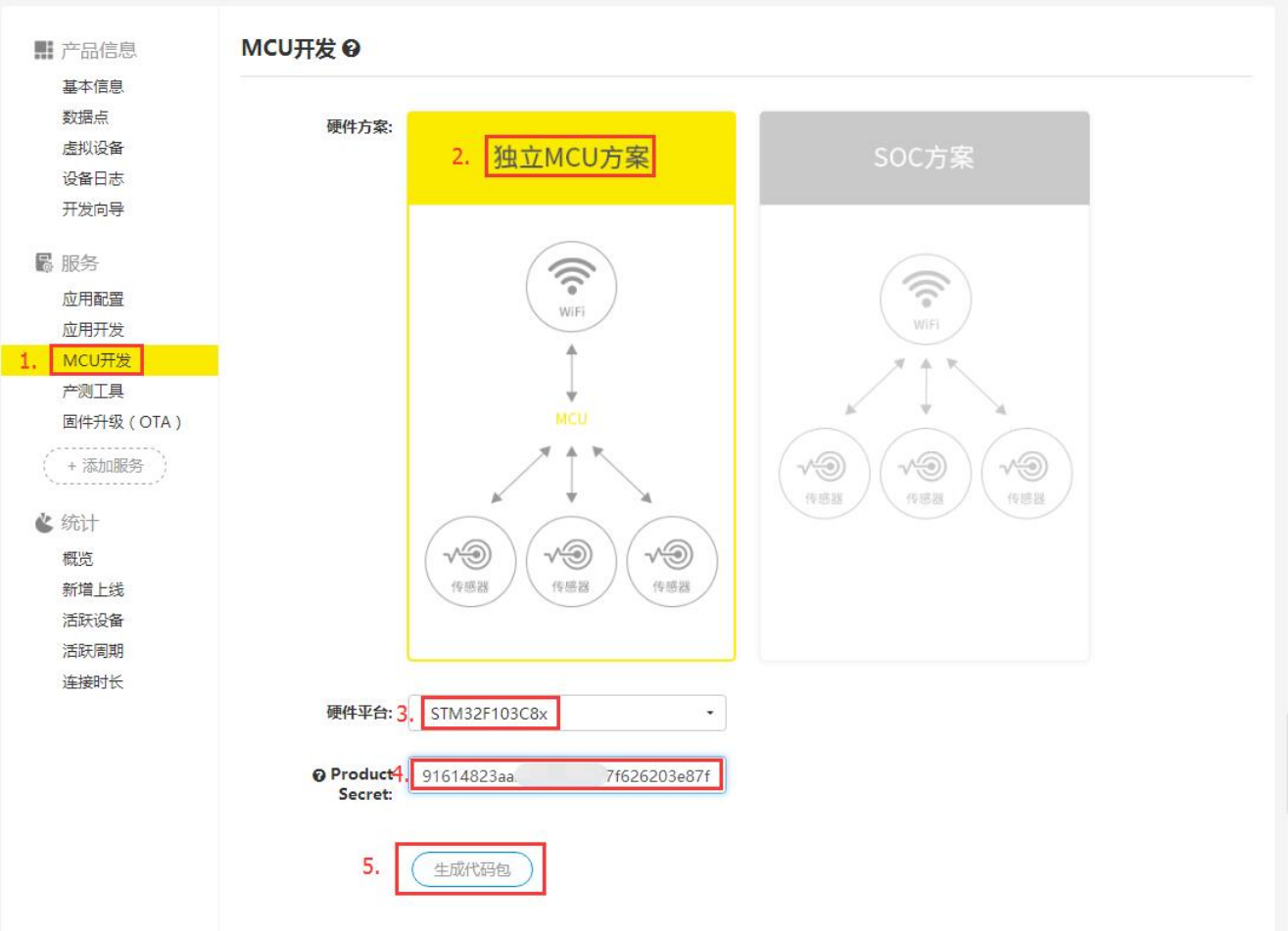
6)可以看到导入的“微信宠物屋”的相关数据点

3 生成微信宠物屋代码

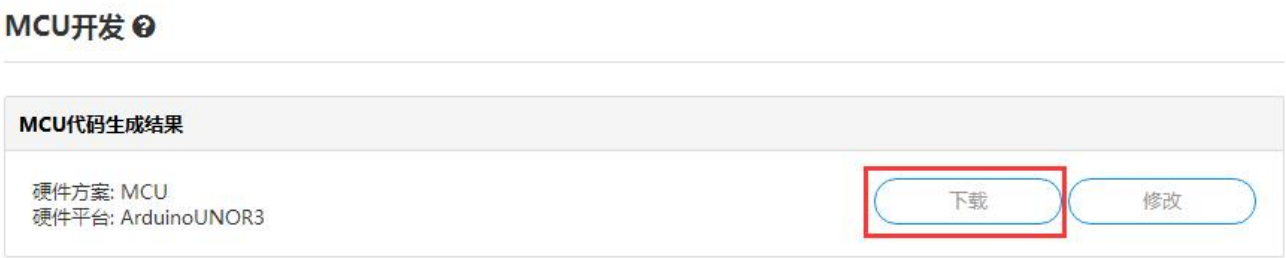
1) 首先，在使用“代码自动生成工具”前要获取产品所对应的**“Product Secret”**（后文简称“PS”）

2) 生成MCU版代码

在“服务”中选择**“MCU方案”**，平台选择为**"STM32F103C8x"**，填入之前在“基本信息”中获得的 **PS**，然后点击“生成代码包”，等待一段时间后便会生成相应的源码工程。

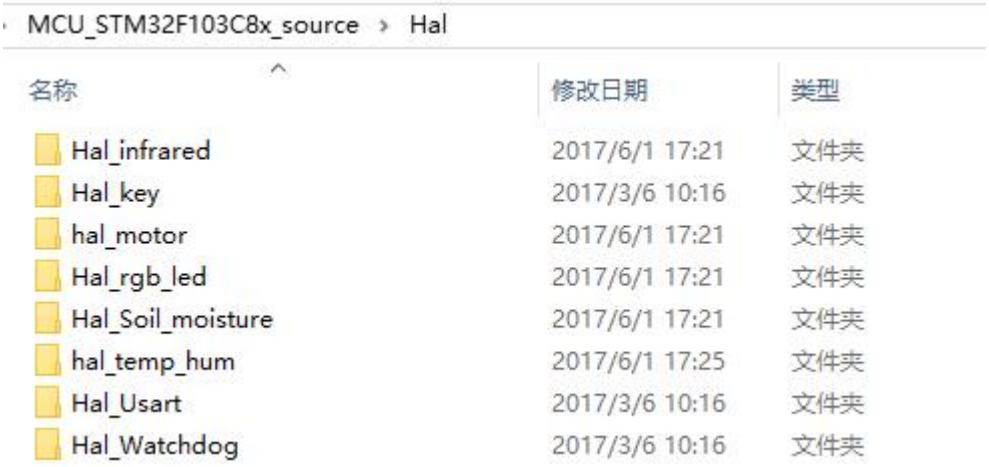


3) 下载生成好的代码工程压缩包并解压

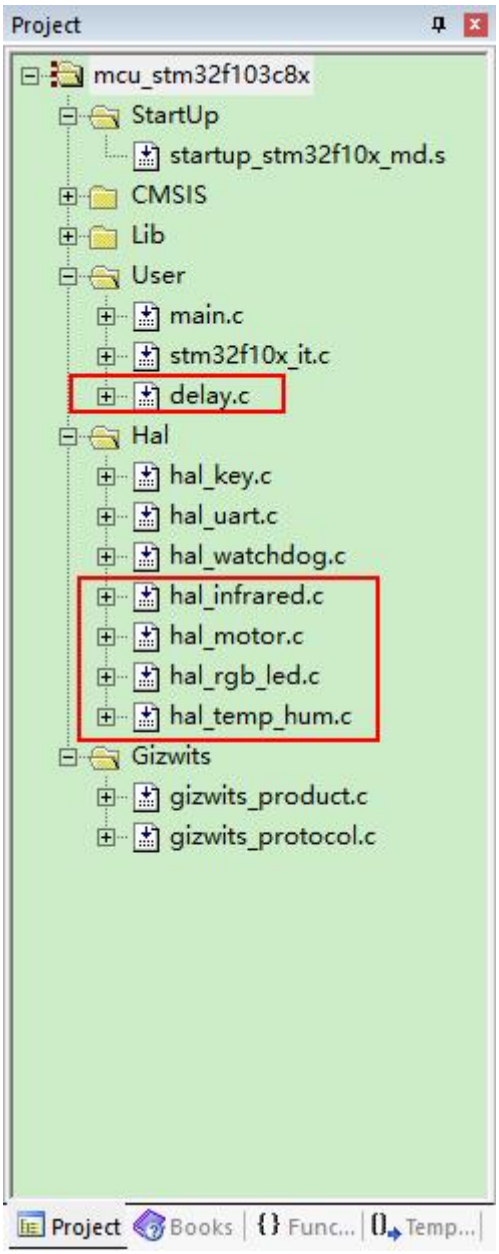


4 移植微信宠物屋驱动代码

1)将 微信宠物屋教程**\STM32**驱动库代码 中的驱动库文件拷贝到自动生成代码工程中的 **MCU_STM32F103C8x_source\Hal** 文件夹中



2)在工程项目中国添加驱动代码的**".c"**文件



3)在代码中添加相应的函数调用

在**MCU_STM32F103C8x_source\User\main.c** 文件中添加各驱动库的头文件

```
#include "Hal_motor/Hal_motor.h"
#include "Hal_rgb_led/Hal_rgb_led.h"
#include "Hal_temp_hum/Hal_temp_hum.h"
#include "Hal_infrared/Hal_infrared.h"
#include "gizwits_product.h"
```

在**MCU_STM32F103C8x_source\User\main.c**文件的**userInit( )**函数中添加各sensor的初始化

```
void userInit(void)
{
    uartxInit();
        watchdogInit(2);
    memset((uint8_t*)&currentDataPoint, 0, sizeof(dataPoint_t));

    delayInit(72);
    rgbLedInit();       ///< 新添加代码: RGB LED初始化
    rgbKeyGpioInit();   ///< 新添加代码: RGB LED初始化
    motorInit();        ///< 新添加代码: 电机初始化
    dht11Init();        ///< 新添加代码: 温湿度初始化
    irInit();           ///< 新添加代码: 红外初始化
    motorStatus(0);     ///< 新添加代码: 电机初始化
}
```

在**MCU_STM32F103C8x_source\User\main.c**文件的**userHandle( )**函数中添加只读型传感器数据点相关的代码

```
void userHandle(void)
{
    uint8_t ret  = 0;
    static uint32_t thLastTimer = 0;

        ///< 新添加代码: 红外传感器数据获取
    currentDataPoint.valueInfrared = irHandle();

        ///< 新添加代码: 温湿度传感器数据获取
    if((gizGetTimerCount()-thLastTimer) > SAMPLING_TIME_MAX)     //上报间隔2S
    {
        ret = dht11Read((uint8_t *)&currentDataPoint.valueTemperature, (uint8_t
*)&currentDataPoint.valueHumidity);
        if(ret != 0)
        {
            printf("Failed to read DHT11\r\n");
        }

        thLastTimer = gizGetTimerCount();
    }
}
```

在**MCU_STM32F103C8x_source\User\main.c**文件的**key2ShortPress( )**函数与**key2LongPress( )**函数中添加长/短按key2时的LED点亮代码

```
void key2ShortPress(void)
{
    printf("KEY2 PRESS ,Soft AP mode\n");

    ///< 新添加代码: Soft AP mode, RGB red
    ledRgbControl(250, 0, 0);

    gizwitsSetMode(WIFI_SOFTAP_MODE);
}
```

```
void key2LongPress(void)
{
    //AirLink mode
    printf("KEY2 PRESS LONG ,AirLink mode\n");

    ///< 新添加代码: AirLink mode, RGB Green
    ledRgbControl(0, 250, 0);

    gizwitsSetMode(WIFI_AIRLINK_MODE);
}
```

在**MCU_STM32F103C8x_source\Gizwits\gizwits_product.c**文件的**gizwitsEventProcess( )**函数中添加可写型传感器数据点相关的代码

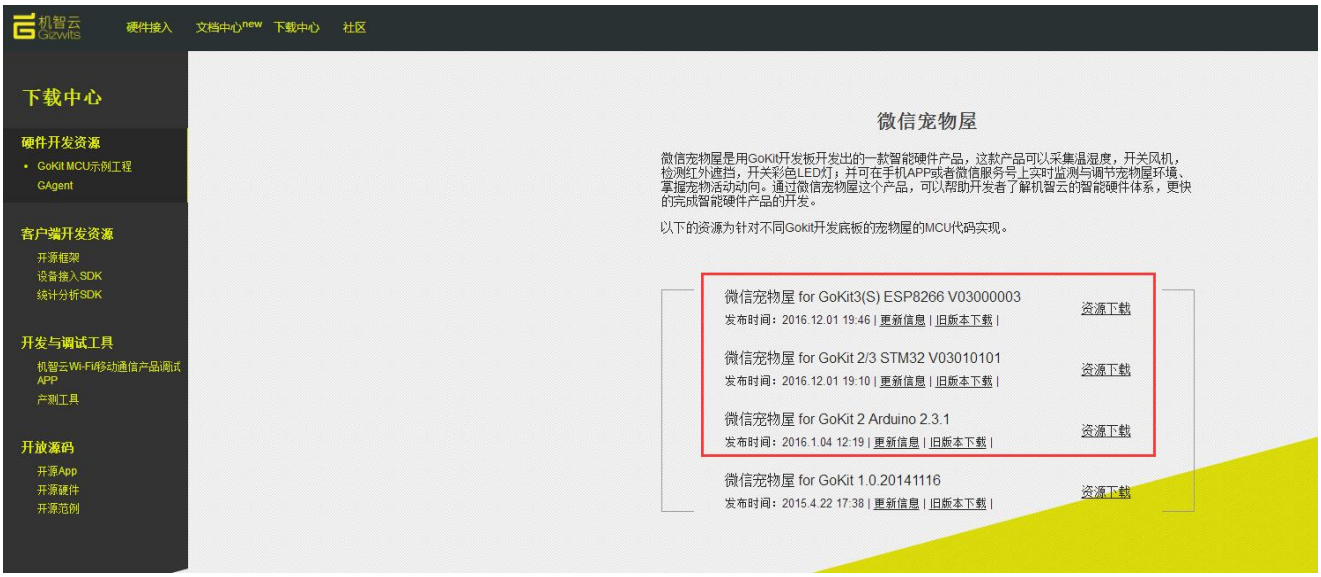
```
int8_t gizwitsEventProcess(eventInfo_t *info, uint8_t *data, uint32_t len)
{
  uint8_t i = 0;
  dataPoint_t *dataPointPtr = (dataPoint_t *)data;
  moduleStatusInfo_t *wifiData = (moduleStatusInfo_t *)data;

  protocolTime_t *ptime = (protocolTime_t *)data;

  if((NULL == info) || (NULL == data))
  {
    return -1;
  }

  for(i=0; i<info->num; i++)
  {
    switch(info->event[i])
    {
      case EVENT_LED_ONOFF:
        currentDataPoint.valueLED_OnOff = dataPointPtr->valueLED_OnOff;
        GIZWITS_LOG("Evt: EVENT_LED_ONOFF %d \n", currentDataPoint.valueLED_OnOff);
        if(0x01 == currentDataPoint.valueLED_OnOff)
        {
            ledRgbControl(254,0,0);         ///< 新添加代码: 对应开启红灯
        }
        else
        {
            ledRgbControl(0,0,0);           ///< 新添加代码: 对应关闭红灯
        }
        break;

      case EVENT_LED_COLOR:
        currentDataPoint.valueLED_Color = dataPointPtr->valueLED_Color;
        GIZWITS_LOG("Evt: EVENT_LED_COLOR %d\n", currentDataPoint.valueLED_Color);
        switch(currentDataPoint.valueLED_Color)
        {
          case LED_COLOR_VALUE0:

ledRgbControl(currentDataPoint.valueLED_R,currentDataPoint.valueLED_G,currentDataPoint.valueLED_B)
;
                                                  break;
          case LED_COLOR_VALUE1:
            ledRgbControl(254, 254, 0);     ///< 新添加代码: 对应LED组合颜色-黄色
            break;
          case LED_COLOR_VALUE2:
            ledRgbControl(254, 0, 70);      ///< 新添加代码: 对应LED组合颜色-紫色
            break;
          case LED_COLOR_VALUE3:
            ledRgbControl(238 ,30 ,30);     ///< 新添加代码: 对应LED组合颜色-粉色
            break;
          default:
            break;
        }
        break;
```

```
      case EVENT_LED_R:
        currentDataPoint.valueLED_R = dataPointPtr->valueLED_R;
        GIZWITS_LOG("Evt:EVENT_LED_R %d\n",currentDataPoint.valueLED_R);
        ///< 新添加代码: 对应设置LED组合色

ledRgbControl(currentDataPoint.valueLED_R,currentDataPoint.valueLED_G,currentDataPoint.valueLED_B)
;
                                break;
      case EVENT_LED_G:
        currentDataPoint.valueLED_G = dataPointPtr->valueLED_G;
        GIZWITS_LOG("Evt:EVENT_LED_G %d\n",currentDataPoint.valueLED_G);
        ///< 新添加代码: 对应设置LED组合色

ledRgbControl(currentDataPoint.valueLED_R,currentDataPoint.valueLED_G,currentDataPoint.valueLED_B)
;
                                break;
      case EVENT_LED_B:
        currentDataPoint.valueLED_B = dataPointPtr->valueLED_B;
        GIZWITS_LOG("Evt:EVENT_LED_B %d\n",currentDataPoint.valueLED_B);
        ///< 新添加代码: 对应设置LED组合色

ledRgbControl(currentDataPoint.valueLED_R,currentDataPoint.valueLED_G,currentDataPoint.valueLED_B)
;
                                break;
      case EVENT_MOTOR_SPEED:
        currentDataPoint.valueMotor_Speed = dataPointPtr->valueMotor_Speed;
        GIZWITS_LOG("Evt:EVENT_MOTOR_SPEED %d\n",currentDataPoint.valueMotor_Speed);
        ///< 新添加代码: 对应设定电机转速
        motorStatus(currentDataPoint.valueMotor_Speed);
        break;
      case WIFI_SOFTAP:
        break;
      case WIFI_AIRLINK:
        break;
      case WIFI_STATION:
        break;
      case WIFI_CON_ROUTER:
              ledRgbControl(0, 0, 0);    ///< 新添加代码: 连接路由后关闭LED灯
        break;
      case WIFI_DISCON_ROUTER:
        break;
      case WIFI_CON_M2M:
        break;
      case WIFI_DISCON_M2M:
        break;
      case WIFI_RSSI:
        GIZWITS_LOG("RSSI %d\n", wifiData->rssi);
        break;
      case TRANSPARENT_DATA:
        GIZWITS_LOG("TRANSPARENT_DATA \n");
        //user handle , Fetch data from [data] , size is [len]
        break;
```

```
      case WIFI_NTP:
        GIZWITS_LOG("WIFI_NTP : [%d-%d-%d %02d:%02d:%02d][%d] \n",ptime->year,ptime->month,ptime-
>day,ptime->hour,ptime->minute,ptime->second,ptime->ntp);
        break;
      default:
        break;
    }
  }

  return 0;
}
```

5 编译烧录固件并进行测试

可在 下载中心 中下载对应平台的**微信宠物屋**官方测试固件以及驱动库文件



测试固件位于：**微信宠物屋教程\Arduino\官方成品固件\mcu_stm32f103c8x.hex**

MCU版的固件烧录方式可查看 ST底板程序编译及下载教程 一文。

微信宠物屋的操作方式可查看 GoKit使用说明书 一文

6 相关说明

> 说明：SOC版与MCU版的工程环境搭建与代码细节介绍请查看文档中心中设备开发的**Gokit**资料一节，本章节只介绍微信宠物屋驱动程序的移植方法。


机智云 Gizwits
文档中心
平台概述
快速入门
APP开发
微信开发
1.
设备开发
云平台
用户手册
English
GAgent详解
MCU开发框架使用说明
2.
Gokit资料
调试与烧写
ECE Demo开发教程
ArduinoUno开发教程
Gokit3系列开发套件简介
Gokit3硬件手册
GoKit3(S)使用说明书
Gokit3(S)开发套件介绍
GoKit3(S)开发环境搭建
Gokit3(S)源码详解
GoKit3(V)使用说明书
Gokit3(V)开发指南
GoKit3(V)词条管理
Gokit3(V)源码详解
Gokit2 使用指南
目录
机智云平台概述
机智云平台架构
机智云优势
机智云全球联网报告
展开全部
平台概述
机智云平台概述
机智云平台是机智云物联网公司经过多年行业内的耕耘及对物联网行
个人、企业开发者的一站式智能硬件开发及云服务平台。平台提供了
试、应用开发、产测、云端开发、运营管理、数据服务等覆盖智能硬
服务的能力。
机智云平台为开发者提供了自助式智能硬件开发工具与开放的云端服
完善的SDK与API服务能力最大限度降低了物联网硬件开发的技术门
提升开发者的产品投产速度，帮助开发者进行硬件智能化升级，更好的连接、服务最终消费者。
机智云平台架构