**AIOT平台固件通用SDK接口文档**

**V01.00.00**

**公司：深圳市泽宝网络科技有限公司**

**2021年6月22**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 文档版本号 | V01.00.00 | | | 保密等级 | 机密 |
| 概要需求文档版本号 | V01.00.00 | | | | |
| 产品文档版本号 | V01.00.00 | | | | |
| 测试用例文档版本号 |  | | | | |
| 代码版本号 | V1.0.0.0 | | | | |
| 产品线 | AIOT | | 归属部门/项目 | | IOT研发部 |
| 编 制 | dawson | 审 核 |  | 批 准 |  |
| 日 期 | 2021-06-22 | 日 期 |  | 日 期 |  |

修改历史

|  |  |  |  |
| --- | --- | --- | --- |
| **序号** | **日期** | **3说明** | **责任人** |
| 1 | 2021-06-22 | 创建初稿 | dawson |
| 2 |  |  |  |

目录

[1. 概述 5](#_Toc75348030)

[2. Device SDK接口定义 5](#_Toc75348031)

[2.1. 接口枚举类型定义 5](#_Toc75348032)

[2.1.1. Aiot环境枚举定义说明 5](#_Toc75348033)

[2.1.2. 日志等级枚举 6](#_Toc75348034)

[2.2. http下载回调函数 6](#_Toc75348035)

[2.3. 初始化函数 7](#_Toc75348036)

[2.4. 反初始化函数 7](#_Toc75348037)

[2.5. 加载配置文件 7](#_Toc75348038)

[2.6. 日志设置函数 7](#_Toc75348039)

[2.7. 登陆AIOT平台 8](#_Toc75348040)

[2.8. 登出AIOT平台 8](#_Toc75348041)

[2.9. AIOT平台mqtt连接 8](#_Toc75348042)

[2.10. 向AIOT平台发送一个POST请求 8](#_Toc75348043)

[2.11. 向AIOT平台发送一个GET请求 9](#_Toc75348044)

[2.12. http下载文件 9](#_Toc75348045)

[2.13. 获取http下载任务的下载进度 9](#_Toc75348046)

[2.14. 停止并销毁HTTP下载任务 10](#_Toc75348047)

[2.15. 获取SDK版本信息 10](#_Toc75348048)

[3. DEVICE SDK接入 10](#_Toc75348049)

[3.1. 创建产品 10](#_Toc75348050)

[3.2. MQTT SDK接入示例 10](#_Toc75348051)

# 概述

设备SDK是泽宝网络科技有限公司(SUNVALLEY) IOT研发部研发的IOT固件设备接入泽宝AIOT的轻量级SDK。它提供了https及mqtt方式连接泽宝AIOT平台。通过本SDK， AIOT平台已经颁发了产品/设备key、产品/设备secret、以及设备SN的设备，可以通过设备SDK简单高效地连接到AIOT平台。

# Device SDK接口定义

## 接口枚举类型及结构体定义

### AIOT环境枚举定义说明

1. A**IOT**环境枚举定义

typedef enum{

e\_aiot\_env\_unknown = -1,

e\_aiot\_env\_default = 0,

e\_aiot\_env\_dev = 1,

e\_aiot\_env\_sit = 2,

e\_aiot\_env\_test = 3,

e\_aiot\_env\_demo = 4,

e\_aiot\_env\_pro = 5

} e\_aiot\_env\_type;

1. Aiot环境枚举定义说明

AIOT的设备环境类型可分为开发（dev）、集成（sit）、测试（test），演示（demo）、发布（pro），而不同环境的设备SN定义不同，可以根据设备SN来区分该设备所在的环境。

* e\_aiot\_env\_default

AIOT默认环境，根据设备SN区分该设备所在的AIOT环境。

* e\_aiot\_env\_dev

AIOT设备开发环境，仅供公司内部开发者开发联调使用的环境，该环境由于处于代码开发调试阶段，微服务容易变动，通常会没有那么稳定。

* e\_aiot\_env\_sit:

AIOT设备集成环境，供公司内部跨部门集成测试环境，该环境用于AIOT平台与APP及功能内部开发的固件联调开发，相对dev环境会比较稳定。

* e\_aiot\_env\_test

AIOT设备测试环境，供公司内部与外研项目的集成测试，该环境用于AIOT平台于公司的研发的APP及外研固件的联调开发，相对dev环境会比较稳定。

* e\_aiot\_env\_demo

AIOT设备演示环境，仅供公司工厂演示和验货使用，只有得到重复测试和验证的微服务、设备和APP才会在进入此环境运行，相对于测试和集成环境，他是一个更加稳定的一个演示/验证环境。

* e\_aiot\_env\_prob

AIOT线上设备使用环境，当demo环境的验货通过后，AIOT平台、设备和APP正式发布上线，是线上设备使用的AIOT环境。他的稳定性非常好，只有经过充分验证的服务、设备和APP在会备发布到线上，为用户提供一个稳定可靠、高效快速的用户体验。

### 日志等级枚举

1. 日志等级枚举定义

typedef enum{

e\_log\_level\_verb = 1,

e\_log\_level\_info = 2,

e\_log\_level\_trace = 3,

e\_log\_level\_warn = 4,

e\_log\_level\_error = 5,

e\_log\_level\_disable = 6

} e\_log\_level\_type;

1. 日志等级枚举类型说明

* e\_log\_level\_verb

输出非常详细的日志

* e\_log\_level\_info

输出比较详细日志

* e\_log\_level\_trace

输出简要日志

* e\_log\_level\_warn

输出警告日志

e\_log\_level\_error

输出错误日志

### mqtt服务质量枚举类型

1. mqtt服务质量枚举类型定义

typedef enum{

e\_aiot\_mqtt\_at\_most\_once = 0,

e\_aiot\_mqtt\_at\_least\_once = 1,

e\_aiot\_mqtt\_exactly\_once = 2

} e\_aiot\_mqtt\_qos;

1. mqtt服务质量枚举项说明

* e\_ aiot\_mqtt\_at\_most\_once

尽力而为的服务类型，消息发送最多一次。对时延、可靠性能不提供任何保证。

* e\_ aiot\_mqtt\_at\_least\_once

综合服务模型，消息发送至少一次，即消息可以发送一次或多次

* e\_ aiot\_mqtt\_exactly\_once:

区别服务模型，消息确保只发送一次

### mqtt消息类型枚举

1. mqtt消息类型枚举定义

typedef enum

{

e\_aiot\_mqtt\_msg\_type\_unknown = -1,

e\_aiot\_mqtt\_msg\_type\_get            = 0,

    e\_aiot\_mqtt\_msg\_type\_control          = 1,

    e\_aiot\_mqtt\_msg\_type\_delete           = 2,

    e\_aiot\_mqtt\_msg\_type\_update           = 3,

    e\_aiot\_mqtt\_msg\_type\_reply           = 4,

    e\_aiot\_mqtt\_msg\_type\_event\_report        = 5,

    e\_aiot\_mqtt\_msg\_type\_property\_report = 6

} e\_aiot\_mqtt\_msg\_type;

1. mqtt消息类型枚举项说明

* e\_aiot\_mqtt\_msg\_type\_unknown

未知消息类型

* e\_aiot\_mqtt\_msg\_type\_get

用于获取指定主题的消息

* e\_aiot\_mqtt\_msg\_type\_control

设备控制消息，此类消息通常带有设备控制指令，设备通常是被动接收此消息并执行相应的指令。

* e\_aiot\_mqtt\_msg\_type\_delete

设备移除、属性/事件删除消息，用于删除？

* e\_aiot\_mqtt\_msg\_type\_update

设备控制响应消息

* e\_aiot\_mqtt\_msg\_type\_reply

设备应答消息

* e\_aiot\_mqtt\_msg\_type\_event\_report

事件上报消息，设备设备发生特定事件时上报的事件消息，如设备的绑定、解绑、开机、关机及用户控制消息触发事件等时

* e\_mqtt\_msg\_type\_property\_report

属性/状态上报消息，通常会用于定时发送设备属性/状态信息，用于保持AIOT平台中的影子服务与设备状态之间的信息同步。

### AIOT OTA消息类型枚举

1. **OTA消息类型**枚举定义

typedef enum

{

e\_aiot\_ota\_msg\_type\_error = -1,

e\_aiot\_ota\_msg\_type\_prepared          = 0,

    e\_aiot\_ota\_msg\_type\_connected          = 1,

e\_aiot\_ota\_msg\_type\_download\_progress      = 2,

e\_aiot\_ota\_msg\_type\_download\_complete      = 3,

e\_aiot\_ota\_msg\_type\_ota\_end = 4,

    e\_aiot\_ota\_msg\_type\_reboot         = 5

} e\_aiot\_ota\_msg\_type;

1. OTA消息类型枚举项说明

* e\_aiot\_ota\_msg\_type\_error

OTA升级中遇到错误，wparam：错误辅助说明参数（errno），lparam：错误码

* e\_aiot\_ota\_msg\_type\_prepared

OTA升级已经准备就绪。wparam：0，lparam：0

* e\_aiot\_ota\_msg\_type\_connected

http请求已连接，准备下载OTA包。wparam：0，lparam：0

* e\_aiot\_ota\_msg\_type\_download\_progress

OTA升级包http下载进度消息，wparam：当前下载速度，lparam：当前下载进度

* e\_aiot\_ota\_msg\_type\_download\_complete

OTA升级包下载结束。wparam：0，lparam：0

* e\_aiot\_ota\_msg\_type\_ota\_end

OTA包下载结束

* e\_aiot\_ota\_msg\_type\_reboot

OTA升级完成，准备重启升级后的系统

### OTA下载信息结构体

1. 结构体定义

typedef struct{

long ldownload\_speed;

long ldownload\_bytes;

long ltotal\_bytes;

long lspend\_time\_ms;

} http\_download\_info;

1. 结构体成功变量说明

* ldownload\_speed：当前下载速度，位每秒
* ldownload\_bytes：当前下载已下载字节数
* ltotal\_bytes：总下载字节数
* lspend\_time\_ms：当前http下载消耗时间，单位毫秒

### OTA请求信息结构体

1. 结构体定义

typedef struct{

long battery;

long region;

long version[16];

} aiot\_otas\_info;

1. 结构体成功变量说明

* battery：电池电量
* region：设备所在的区域代码
* version：当前固件版本号

## OTA升级进度回调函数

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | on\_ota\_msg\_cb |
| 2 | 接口说明 | OTA升级进度回调函数，由调用者实现 |
| 3 | 参数 1 [input] | long handle:http下载任务id |
| 4 | 参数 2 [input] | void\* userdata: 下载调用者实例指针 |
| 5 | 参数 3 [input] | long msg\_id：回调消息类型，详见2.1.6 e\_aiot\_http\_msg\_type定义 |
| 5 | 参数 3 [input] | long wparam，根据msg\_id不同， wparam的类型和含义可能不一样，详见2.1.6 http回调消息枚举项说明 |
| 6 | 参数 4 [input] | long lparam： 根据msg\_id不同， lparam的类型和含义可能不一样，详见2.1.6 http回调消息枚举项说明 |
| 8 | 返回值(void) | 无 |

## MQTT事件回调函数

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | on\_mqtt\_event\_cb |
| 2 | 接口说明 | MQTT事件回调通知函数，由应用层实现 |
| 3 | 参数 1 [input] | long handle:AIOT实例句柄 |
| 4 | 参数 2 [input] | const char\* base：字符串信息 |
| 5 | 参数 3 [input] | int event\_id: 事件id |
| 6 | 参数 4 [input] | void\* event\_data:事件数据结构体，详见乐鑫mqtt\_client的esp\_mqtt\_event\_t结构体定义 |
| 7 | 返回值(int) | 成功返回0，否则失败 |

## MQTT订阅主题回调函数

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | on\_mqtt\_subscribe\_cb |
| 2 | 接口说明 | MQTT订阅主题回调通知函数，由应用层实现 |
| 3 | 参数 1 [input] | long handle:AIOT实例句柄 |
| 4 | 参数 2 [input] | void\* userdata: MQTT订阅主题调用者的对象指针 |
| 5 | 参数 3 [input] | const char\* ptopic: MQTT回调消息的订阅主题 |
| 6 | 参数 4 [input] | void\* payload: 订阅消息的负载内容 |
| 7 | 参数 5 [input] | int payload\_len: 订阅消息负载的长度（字节） |
| 8 | 参数 6 [input] | e\_mqtt\_qos qos: 消息服务质量，详见2.1.1 |
| 9 | 返回值(int) | 成功返回0，否则失败 |

## 初始化函数

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_init |
| 2 | 接口说明 | 创建AIOT实例句柄 |
| 3 | 参数 1 [input] | const char\* client\_id：租户id/产品key |
| 4 | 参数 2 [input] | const char\* client\_secret：租户密钥 |
| 5 | 参数 3 [input] | const char\* device\_sn：设备SN |
| 6 | 参数 4 [input] | const char\* cer\_path：证书路径 |
| 6 | 参数 4 [input] | e\_aiot\_env\_type env:AIOT环境，详见2.1.1，默认为 e\_aiot\_env\_default：根据设备SN特征区分环境 |
| 7 | 返回值(void\*) | 成功返回AIOT实例句柄，失败返回NULL |

## 反初始化函数

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_deinit |
| 2 | 接口说明 | 反初始化设备sdk，销毁全局变量、关闭日志及配置文件上下文 |
| 3 | 参数 1 [input] | void\* handle: AIOT实例句柄 |
| 7 | 返回值(void) | 无 |

## 加载配置文件

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_load\_config |
| 2 | 接口说明 | 加载SDK配置文件 |
| 3 | 参数 1 [input] | const char\*: purl：配置文件加载路径 |
| 4 | 返回值(void) | 成功返回0，否则为失败 |
| 5 | 备注 | 此接口支持从服务器或本地加载配置文件，服务端目前暂未能支持 |

## 日志设置函数

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_init\_log |
| 2 | 接口说明 | 设置日志等级，输出模式及输出路径 |
| 3 | 参数 1 [input] | level，日志等级，详见2.1.2日志等级枚举定义 |
| 4 | 参数2 [input] | mode，日志输出模式，1：控制台输出日志，2：文件输出日志，3：控制台&文件同时输出日志，4：输出到aiot平台 |
| 5 | 参数3 [input] | path：日志输出路径。仅当mode&2为真的情况下路径设置才会生效，如果路径设置为NULL，就算mode&2为真也不会输出日志 |
| 6 | 返回值(int) | 成功返回0，失败返回非零 |

## 连接AIOT平台

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_connect |
| 2 | 接口说明 | 创建aiot平台连接 |
| 3 | 参数 1 [input] | aiot实例句柄 |
| 4 | 返回值(long) | 成功返回0，否则为失败 |
| 5 | 备注 | AIOT连接包括建立MQTT和获取、刷新access\_token |

## 断开AIOT平台连接

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_disconnect |
| 2 | 接口说明 | 断开SDK与AIOT平台的连接 |
| 3 | 参数 1 [input] | Long handle: AIOT实例句柄 |
| 4 | 返回值(void) | 无 |

## 向AIOT平台发送一个POST请求

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_http\_post |
| 2 | 接口说明 | 使用http发送一个post请求 |
| 3 | 参数1 [input] | long handle: AIOT实例句柄 |
| 4 | 参数2 [input] | const char\* path：http请求路径 |
| 5 | 参数3 [input] | const char\* req：http请求消息体（默认为jason） |
| 6 | 参数4 [output] | int\* phttp\_code：http状态码 |
| 7 | 参数5 [output] | const char\* resp: http post请求的响应内容接收buffer |
| 8 | 参数6 [input] | int len：接收http响应的buffer长度 |
| 9 | 返回值(int) | 成功返回http响应内容的长度，否则为失败 |
| 10 | 备注 | 设备SDK的AIOT域名默认为https://iot-api{$env}.sunvalley.com |

## 创建OTA升级任务

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_create\_oat\_task |
| 2 | 接口说明 | 创建OTA升级任务 |
| 3 | 参数1 [input] | long handle: AIOT实例句柄 |
| 4 | 参数2 [input] | aiot\_ota\_info\* pota\_info：ota升级信息结构体指针 |
| 5 | 返回值(long) | 成功返回OTA任务id，失败返回0 |
| 6 | 备注 | 该函数将会创建一个线程去执行OTA任务 |

## 设置OTA升级任务回调函数

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_set\_ota\_callback |
| 2 | 接口说明 | 设置OTA升级任务回调函数 |
| 3 | 参数1 [input] | void\* task\_id：ota升级任务id |
| 4 | 参数2 [input] | on\_ota\_msg\_cb ota\_cb：升级任务回调函数 |
| 5 | 参数3[input] | void\* owner：调用者指针 |
| 6 | 返回值(int) | 成功返回0，否则为失败 |

## 开始OTA升级任务

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_ ota\_start |
| 2 | 接口说明 | 开始ota升级任务 |
| 3 | 参数1 [input] | void\* task\_id：ota升级任务id |
| 4 | 返回值(int) | 成功返回0，否则为失败 |

## 获取OTA任务进度

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_ota\_get\_progress |
| 2 | 接口说明 | 获取当前http下载任务的下载进度 |
| 3 | 参数1 [input] | void\* task\_id：ota升级任务id |
| 4 | 参数2 [input] | long task\_id：http下载任务id |
| 5 | 参数3 [output] | e\_aiot\_ota\_msg\_type\* msg\_id：http下载进度消息类型，详见2.1.5定义 |
| 6 | 参数4 [output] | http\_download\_info\* phdp：HTTP下载进度信息，详见2.1.6定义 |
| 7 | 参数5 [output] | int\* percent，当前http下载进度（百分比） |
| 8 | 返回值(int) | 成功返回0，失败返回非0值(-1) |

## 停止并销毁OTA任务

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_ota\_stop |
| 2 | 接口说明 | 停止并销毁ota任务 |
| 3 | 参数 [input] | void\* task\_id：OTA任务id |
| 4 | 返回值(int) | 成功返回0，失败返回非0值(-1) |

## 获取SDK版本信息

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_version |
| 2 | 接口说明 | 获取AIOT SDK版本信息 |
| 3 | 参数1 [input] | const char\* szver：版本信息接收内存buffer指针 |
| 4 | 参数2 [input] | int len：版本信息接收内存buffer长度 |
| 5 | 返回值(int) | 成功返回0，失败返回非0值(-1) |

## mqtt发布消息

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_mqtt\_pub |
| 2 | 接口说明 | 向一个主题发布消息 |
| 3 | 参数1 [input] | void\* handle: aiot 实例句柄 |
| 4 | 参数2 [input] | const char\* ptopic: 发布消息的主题，可以为NULL |
| 5 | 参数3 [input] | const char\* payload: 发布消息的内容 |
| 6 | 参数4 [input] | e\_mqtt\_qos qos: 发布消息的服务质量，详见2.1.3 |
| 7 | 返回值(int) | 成功返回0，失败返回非0值(-1) |
| 8 | 备注 | 发布主题可以为NULL，SDK默认为： /shadow/${product\_key}/${device\_sn}/update |

## 发布控制指令消息

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_mqtt\_pub\_ctrl |
| 2 | 接口说明 | 向给定主题发布一个设备控制消息 |
| 3 | 参数1 [input] | void\* handle: aiot 实例句柄 |
| 4 | 参数3 [input] | const char\* command: 发布指令消息 |
| 5 | 参数4 [input] | e\_mqtt\_qos qos: 发布消息的服务质量，详见2.1.3 |
| 6 | 返回值(int) | 成功返回0，失败返回非0值(-1) |
| 7 | 备注 | 本接口只发送指令控制（e\_mqtt\_msg\_type\_control）消息  发布主题默认为： /shadow/${product\_key}/${device\_sn}/update |

## 发布控制指令响应消息

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_mqtt\_pub\_update |
| 2 | 接口说明 | 向给定主题发布一个设备控制响应消息 |
| 3 | 参数1 [input] | void\* handle: aiot 实例句柄 |
| 4 | 参数2 [input] | const char\* state: 设备状态/属性消息（"state"内容， json格式） |
| 5 | 参数3 [input] | const char\* result: 指令执行结果（"result"内容， json格式） |
| 6 | 参数4 [input] | e\_mqtt\_qos qos: 发布消息的服务质量，详见2.1.3 |
| 7 | 返回值(int) | 成功返回0，失败返回非0值(-1) |
| 8 | 备注 | 发布主题默认为： /shadow/${product\_key}/${device\_sn}/update |

## 发布事件上报消息

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_mqtt\_pub\_event |
| 2 | 接口说明 | 向给定主题发布一个事件上报消息 |
| 3 | 参数1 [input] | void\* handle: aiot 实例句柄 |
| 4 | 参数2 [input] | const char\* event: 事件名称 |
| 5 | 参数3 [input] | const char\* data: 事件内容（"data"内容， json格式） |
| 6 | 参数4 [input] | e\_mqtt\_qos qos: 发布消息的服务质量，，详见2.1.3 |
| 7 | 返回值(int) | 成功返回0，失败返回非0值(-1) |
| 8 | 备注 | 发布主题默认为： /shadow/${product\_key}/${device\_sn}/update |

## 发布属性上报消息

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_mqtt\_pub\_property |
| 2 | 接口说明 | 向给定主题发布一个事件上报消息 |
| 3 | 参数1 [input] | void\* handle: aiot 实例句柄 |
| 4 | 参数2 [input] | const char\* ptopic: 发布消息的主题 |
| 5 | 参数3 [input] | const char\* state: 属性状态消息（"state"内容， json格式） |
| 6 | 参数4 [input] | e\_mqtt\_qos qos: 发布消息的服务质量，，详见2.1.3 |
| 7 | 返回值(int) | 成功返回0，失败返回非0值(-1) |
| 8 | 备注 | 发布主题默认为： /shadow/${product\_key}/${device\_sn}/update |

## 订阅消息

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_mqtt\_sub |
| 2 | 接口说明 | 订阅指定主题的消息 |
| 3 | 参数1 [input] | void\* handle: aiot 实例句柄 |
| 4 | 参数2 [input] | const char\* ptopic: 订阅消息的主题 |
| 5 | 参数3 [input] | e\_mqtt\_qos qos: 发布消息的服务质量，详见2.1.3 |
| 6 | 参数4 [input] | on\_mqtt\_message\_cb on\_msg: 接收到订阅消息的回调处理函数，详见2.3定义 |
| 7 | 参数5 [input] | Void\* userdata: 订阅消息的拥有者的指针，在回调函数中作为回调消息的拥有者传递给被调用者 |
| 8 | 返回值(int) | 成功返回0，失败返回非0值(-1) |

## 取消订阅

| **No** | **信息** | **描述** |
| --- | --- | --- |
|  |  |  |
| 1 | 接口名称 | sun\_aiot\_api\_mqtt\_unsub |
| 2 | 接口说明 | 取消订阅给定主题的消息 |
| 3 | 参数1 [input] | void\* handle: aiot 实例句柄 |
| 4 | 参数2 [input] | const char\* ptopic: 订阅消息的主题 |
| 5 | 返回值(int) | 成功返回0，失败返回非0值(-1) |

# 设备SDK接入

## 创建产品

在使用DEVICE SDK之前，你必须在泽宝AIOT平台创建自己的产品，创建完成后会生成对应的产品ID(product\_id)和产品密钥(product\_secret)，并创建该产品的设备的序列号（device\_sn），AIOT平台的设备认证方式分为一机一密和一型一密两种。如果是一机一密的认证方式，创建设备SN的同时还会生成对应的设备key和设备密钥。

## 设备SDK接入示例

### 连接前准备工作

在泽宝AIOT平台创建完自己的产品和设备后，你会得到以下信息：

* mqtt\_host：MQTT服务域名
* mqtt\_port:9903, MQTT服务端口号
* Product\_ID：32位产品id
* Product\_Key：32位产品key
* Product\_Secret：32位密钥
* Device\_SN：设备SN
* Device\_Secret：32位设备密钥（一机一密认证独有）

### 设备 SDK使用示例

详细使用示例参加demo