### **概述**

本SDK是基于网关联网SDK进行封装而来，主要用于Android系统。

本SDK使用需要开启读写权限。

本SDK通过IoTGatewaySDKManager类对外提供主动调用和回调两部分接口。

下面从这两方面介绍下使用说明。

### **主调接口**

IoTGatewaySDKManager getInstance()

**功能说明**

获取SDK管理器实例。

**参数说明**

无

**返回值**

SDK管理器实例

**int** IotGatewayStart(Context context, Config config, GwAttachAttr[] attachAttrs)

**功能说明**

启动网关设备。

**参数说明**

|  |  |  |
| --- | --- | --- |
| **参数名称** | **说明** |  |
| context | context of app |  |
| config | 是IoTGatewaySDKManager的内部类，需进行实例化，参考Config |  |
| attachAttrs | 对应联网SDK的GW\_ATTACH\_ATTR\_T结构体 |  |

**返回值**

|  |  |  |
| --- | --- | --- |
| **返回值** | **说明** |  |
| 0 | 操作成功 |  |
| 错误码 | 失败返回错误码 |  |

**int** startFwDownload()

**功能说明**

当有新版本固件通知到用户时，用户通过此函数启动固件下载。固件下载完整路径需已在Config对象中填充传递。

**参数说明**

无

**返回值**

|  |  |  |
| --- | --- | --- |
| **返回值** | **说明** |  |
| 0 | 操作成功 |  |
| 错误码 | 失败返回错误码 |  |

**说明：**

\* 发送dp事件

\* @param id dp id

\* @param type 类型 DPEvent.Type

\* DPEvent.Type.PROP\_BOOL boolean

\* DPEvent.Type.PROP\_VALUE int

\* DPEvent.Type.PROP\_STR string

\* DPEvent.Type.PROP\_ENUM int

\* DPEvent.Type.PROP\_RAW byte[]

\* @param val 值

\* @return

**int** sendDPWithTimeStamp(String devId, **int** dpId, **int** type, Object val, **int** timestamp)

**说明：**

\* 发送dp事件带时间戳

\*

\* @param id dp id

\* @param type 类型 DPEvent.Type

\* @param val 值

\* @param timestamp 时间戳 单位秒

\* @return

**int** sendDP(String devId, DPEvent... events)

**说明：**

\* 发送多个dp事件

\*

\* @param events 多个dp类型

\* @return

**int** sendDPWithTimeStamp(String devId, DPEvent... events)

**说明：**

\* 发送多个dp事件带时间戳（时间戳需要赋值在DPEvent.timestamp）

\*

\* @param events 多个dp类型

\* @return

**int** setGwEngrLogPath(String path)

工程模式日志存储路径设置。

**int** setGwAppLogPath(String path)

### **回调接口**

回调接口分为4大类：IoTAppCallbacks、IoTCallbacks、IoTGwCallbacks、UpgradeEventCallback

IoTAppCallbacks

用户实现此接口后，通过**void** setIoTAppCallbacks(IoTAppCallbacks callbacks)接口注册到管理器中。

**额外接口说明：**

**void** onGwEngrToNormalFinish(String path);

IoTCallbacks

**说明：**

1. 升级相关接口无需关注，已挪到UpgradeEventCallback中维护。
2. 3个dp回调接口统一通过onDpEvent接口回调回来。

3.另外，加入了几个额外接口，如下：

**int** onNetworkStatus(**int** status); *// GW外网状态变动回调*

status值：

/\* offline in LAN.  user wired callback <tuya\_hal\_wired\_station\_conn> return <false> \*/

#define GB\_STAT\_LAN\_UNCONN 0

/\* online in LAN, offline in WAN.

   user wired callback <tuya\_hal\_wired\_station\_conn> return <true> but mqtt is offline

\*/

#define GB\_STAT\_LAN\_CONN 1

/\* online in WAN.

   user wired callback <tuya\_hal\_wired\_station\_conn> return <true> and mqtt is online

\*/

#define GB\_STAT\_CLOUD\_CONN 2

*/\* 工程模式使用 \*/*

**int** onEngrGwSetChannel(**int** channel); // 对应gw\_set\_channel\_cb

**int** onEngrGwGetChannel(**int** channel); // 对应gw\_get\_channel\_cb

**int** onEngrGwGetLog(**int** lengthLimit); // 对应gw\_get\_log\_cb

**int** onEngrGwSyncConfig(); // 对应gw\_sync\_config\_cb

**int** onEngrGwEngineerFinish(); // 对应gw\_engineer\_finish\_cb

IoTGwCallbacks

用户实现此接口后，通过**void** setIoTGwCallbacks(IoTGwCallbacks callbacks)接口注册到管理器中。

**说明：**

无

UpgradeEventCallback

升级事件回调单独设立了一个回调接口。需要升级的时候注册回调就行。

接口说明：

*/\*\**

*\* sdk 接收到后端的升级推送的时候，会触发此接口 附带升级信息*

*\** ***@param version***

*\*/*

**void** onUpgradeInfo(String version);

*/\*\**

*\* 升级文件开始下载*

*\*/*

**void** onUpgradeDownloadStart();

*/\*\**

*\* 升级文件下载进度*

*\*/*

**void** onUpgradeDownloadUpdate(**int** progress);

*/\*\**

*\* sdk 下载升级文件下载完成触发此接口*

*\*/*

**void** upgradeFileDownloadFinished(**boolean** success);

*/\*\**

*\* 升级失败*

*\** ***@param msg*** *错误信息*

*\*/*

**void** onUpgradeFail(String msg);

### **辅助类**

**Config类**

重点关注项及说明：

**public final static class** Config {

*//存储路径，注意长度，底层为64个char字符长度的数组，还追加了一个文件名，也就是说路径要比64位还要短。*

**public** String **mPath**;

*//固件key*

**public** String **mFirmwareKey**;

*//uuid和authkey成对出现，而且设备唯一*

**public** String **mUUID**;

**public** String **mAuthKey**;

*//固件版本号 & 包名*

**public** String **mVersion**;

**public** String **mPackageName**;

*//网关是否具有设备的属性*

**public boolean mGwAsDev**;

*// 是否工程模式*

**public boolean mEngrMode**;

...

**AlarmInfo类**

**public class** AlarmInfo {

**public** String **alarm\_mode**; *// 32byte max*

**public byte alarm\_status**;

**public byte enable\_countdown\_status**;

...

**GwAttachAttr类**

**public class** GwAttachAttr {

*/\* tp value list: \*/*

**public static final byte *DEV\_BLE\_SNGL*** = 1; *//BLE固件*

**public static final byte *DEV\_ZB\_SNGL*** = 3; *//ZigBee固件*

**public static final byte *DEV\_NM\_NOT\_ATH\_SNGL*** = 9; *//MCU固件*

**public byte tp**;

**public** String **ver**;

...

**ChCode类**

**public class** ChCode {

**public static final int *CH\_CODE\_LMT*** = 15;

**public static final int *CH\_SN\_LMT*** = 20;

**public static final int *CH\_REPORT\_CODE\_LMT*** = 20;

**public static final int *CH\_MANU\_ID\_LMT*** = 10;

**public static final int *CH\_VERSION\_LMT*** = 10;

**public static final int *CH\_ENCRYPT\_KEY\_LMT*** = 20;

**public** String **ch\_name**;

**public** String **ch\_code**;

**public** String **ch\_sn**;

**public** String **ch\_report\_code**;

**public** String **ch\_manu\_id**;

**public** String **ch\_version**;

**public** String **ch\_encrype\_key**;

...

**DP事件类：**

*/\*\**

*\* DP点类*

*\*/*

**public class** DPEvent {

**public class** Type {

*//Boolean*

**public static final int *PROP\_BOOL*** = 0;

*//Integer*

**public static final int *PROP\_VALUE*** = 1;

*//String*

**public static final int *PROP\_STR*** = 2;

*//Integer*

**public static final int *PROP\_ENUM*** = 3;

*//Integer*

**public static final int *PROP\_BITMAP*** = 4;

*//RAW*

**public static final int *PROP\_RAW*** = 5;

}

**public int dpid**; *// dp id*

**public short type**; *// dp type*

**public** Object **value**; *// dp value*

*/\*\**

*\* 发生的时间戳(单位秒)*

*\*/*

**public int timestamp**; *// dp happen time. if 0, mean now*

...