STP SDK 说明文档

（Android）

一、SDK 概述

1. 模块介绍

本 SDK 主要提供了智能点读笔常用的功能接口。本文档会对接口按照模块进行分类和使用说明，以及集成、调用方式等内容。

文档的第二章说明现有项目，如何快速集成 SDK 进行开发。

第三章之后说明各模块具体的功能接口，调用方式及参数说明。

SDK 根据功能可以划分为如下几个模块：

1) 账户模块

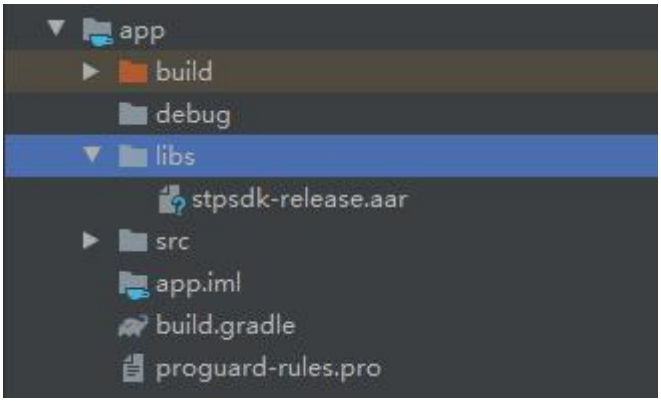
2) 设备管理模块

3) 绘本管理模块

4) 学习记录模块

二、SDK 集成步骤

1. 将 SDK 的 AAR 文件(例如：stpsdk-release.aar) 放入工程的 libs 目录下，如下图所示：



2. 编辑 app 的 build.gradle 文件，添加引用说明，如下图所示：

```
//个推
implementation 'com.getui:sdk:2.14.0.0'
//StpSDK
implementation(name: 'stpsdk-release', ext: 'aar')
```

3. 点击 Gradle Sync 按钮进行刷新后，就可以使用 SDK 了。

三、SDK 整体说明

1. SDK 概述

SDK 功能上分为 4 个大模块，每个模块对应了一个 Manager 类，提供具体业务功能接口。

各模块类名分别是：AccountManager，DeviceManager，StudyReportManager，DeviceSdcardManager。

2. SDK 初始化

每个 SDK 客户会分配一个不同的 packageId，用于区分不同客户。SDK 初始化只要填写 packageId 信息即可。

建议在项目的第一个 Activity 的 OnCreate()内执行初始化。例如：

```
@Override
public void onCreate() {
    super.onCreate();
    initStpSDK();
}

private void initStpSDK() {
    StpSDK.getInstance().init(this, "qpy.sdk");
}
```

返回结果说明

SDK 的接口分为同步和异步两种调用类型，一般本地可以快速返回的接口是同步调用接口；需要等待的都是异步调用接口，如各种网络请求接口，所有网络请求的回调都是 ResultListener 类型。

ResultListener 定义：

```
public abstract class ResultListener {
    public abstract void onSuccess(Result result);

    public abstract void onError(int errCode, String errMsg);
}
```

onSuccess: 表示接口请求成功，但是不代表业务功能执行成功。例如：调用登录接口，网络请求正常，但是密码填写错误，依然会回调 onSuccess，应该通过判定 result.getResult() 值，来判断是否登录成功。

onError: 表示接口请求出错，例如出现超时，异常等问题

输出结果统一是 json 格式，由{“result”,”msg”，”data”}3 个字段组成。当返回错误时，部分接口有“desc”字段提供更详细的错误描述信息。

错误返回举例：

```
{
    "result": -111,
        "msg": "passwd is not match with phone!",
        "desc": "require encrypted
password(2ea70c7e3c62c440db6ea392e8a86df4), actual is
(2139aedcf7ac4750157acd6f63ecb9a8)"
}
```

返回结果可以分 3 种类型分别对应：

1） 无数据类型（无 data 字段）

```
{
   "result": 0,
   "msg": “succ”
}
```

2) 对象数据类型（data 字段是一个 JsonObject）

```
{
   "result": 0,
   "msg": “succ”,
   “data” : { .... }
}
```

3) 数组数据类型（data 字段是一个 JsonArray）

```
{
   "result": 0,
   "msg": “succ”,
   “data” : [ .... ]
}
```

四、账户模块（AccountManager 类）

1. 登录

1.1. 登录方式说明

登录有两种方式：

1) 使用 STP 的账户体系，账户数据保存在 STP 服务器

2) 使用客户原有账户体系，双方云端做账户系统对接，STP 服务器不保存用户账户信息

1.2. 账户系统对接过程

账户系统如果使用双方云端对接方式，用户使用客户的账户体系，我们云端只有登录，不需要提供注册、修改密码、注销等操作。

具体流程：

用户输入账户和密码，登录客户账户系统

客户给我们云端发送账户 ID（AccountId）和 token，token 是客户计算的一个加密字符串（每次登录会不一样）

我们云端将手机号和 token 发送给客户的验证接口，进行鉴权验证，若客户接口返回认证通过，则认为本次登录合法

（若该手机号是第一次登录，我们云端会用该手机号进行注册，并作为内部用户账号）。

反之，若接口返回认证失败，则认为登录违法，返回登录错误

1.3. 接口说明

1) STP 自有账户登录

```
// phone：用户手机号（必选）
// passwd：账户密码（必选）
// pushId： 推送 ID，个推 SDK 初始化后会获得该 ID (可选)
public void login(String accountId, String passwd, String pushId, ResultListener listener)
```

返回结果：

参看：com.aiedevice.sdk.account.bean.LoginData

2) 客户账户对接登录

```
// userId ： 客户给用户账户对应生成的账户 id，也可以直接填用户账户信息 （必选）
// thirdCode ： 针对用户本次登录生成的验证 token（必选）
public void loginEx(String userId, String thirdCode, ResultListener listener)
```

返回结果：

参看：com.aiedevice.sdk.account.bean.LoginData

2. 设置 pushId

2.1. 接口说明

在登录的时候可以填写 pushid，若登录时未填写，可以之后执行该接口，更新 pushid

1) 设置 pushid

```
// pushId: 个推 SDK 初始化后返回的 pushId（必选）
public boolean setPushId(String pushId, ResultListener listener)
```

返回结果：

成功：

```
{
    "result": 0,
    "msg": "success"
}
```

失败：

```
{
    "result": errCode,
    "msg": errMessage
}
```

3. 注销

3.1. 接口说明

1) 注销

```
public boolean logout(ResultListener listener)
```

返回结果：

成功：

```
{
    "result": 0,
    "msg": "success"
}
```

失败：

```
{
    "result": errCode,
    "msg": errMessage
}
```

4. 选定当前设备

4.1. 功能说明

设定当前设备 ID，之后相关设备操作，以该设备为操作对象

4.2. 接口说明

1) 选定设备

public boolean setDeivceId(string deviceId);

返回结果：

成功：true

失败：false

五、设备管理模块（DeviceManager 类）

1. 配网和绑定

1.1. 功能说明

当用户第一次使用点读笔时，下载 APP、注册登录账户后，就会进行配

网和绑定操作。该过程是一个原子操作，一处失败就认为整体失败。具体配网和绑定流程如下：

1) 手机检测蓝牙是否开启，没有则开启蓝牙
2) 提示用户将设备置位待配网状态
3) 扫描设备
4) 扫描并选择 SSID，然后输入 Wifi 密码
5) 发送 Wifi 配置和用户信息给设备
6) 手机端轮询服务器，检查设备是否上报绑定信息
7) 若收到绑定信息则配网完成，否则超时后，报告配网失败

1.2. 接口说明

如上一节配网和绑定流程所示，第 1 步和 4 步操作，由 demo 代码负责实现，SDK 负责实现第 5 步到第 7 步。

1) 配置网络和绑定

```
// context: 上下文（必选）
// device : 蓝牙设备对象（必选）
// params： Wifi 配置参数（必选）
public boolean startConfigure(Context context, BluetoothDevice device, BlufiConfigureParams
                              params, ResultListener listener)
```

返回结果：

参看：com.aiedevice.bean.Result

成功：

```
{
    "result": 0,
```

```
    "msg": "success"
}
```

失败：

```
{
    "result": errCode,
    "msg": errMessage
}
```

2. 解除绑定

2.1. 接口说明

1) 解除绑定

public void unbindDeivce(ResultListener listener);

返回结果：

参看：com.aiedevice.bean.Result

成功：

```
{
    "result": 0,
    "msg": "success"
}
```

失败：

```
{
    "result": errCode,
    "msg": errMessage
}
```

3. 获取设备列表

3.1. 接口说明

1) 获取设备列表

// isDetail: 是否获取设备详细信息（必选）

// true： 获取详细信息，如：有哪些用户绑定该设备
// false：获取概要信息
public void getDeviceList(bool isDetail, new ResultListener()

返回结果：

参看：com.aiedevice.sdkdemo.bean.MainctrlListData

4. 修改设备备注名

4.1. 接口说明

1) 修改备注名

// newDeviceName: 修改后的设备名（必选）
public void updateDeviceName(string newDeviceName, ResultListener listener);

返回结果：

```
{
    "result": 0,
    "msg": "success"
}
```

失败：

```
{
    "result": errCode,
    "msg": errMessage
}
```

5. 获取设备信息

5.1. 功能说明

通过该接口可以获取设备如下信息和状态：

1) 连接的 Wifi 名称

2) IP 地址

3) MAC 地址

4) 设备类型

5) SN

6) 是否在线

7) 获取设备名称（备注名）

8) 绑定的人员列表

9) 电量

10) 充电状态

11) 音量大小

5.2. 接口说明

1) 获取设备信息

public void getDeviceDetail(ResultListener listener);

返回结果：

参看：com.aiedevice.basic.bean.DeviceDetail

6. 获取设备硬件信息（新增）

6.1. 接口说明

1) 修改备注名

// newDeviceName: 修改后的设备名（必选）
public void getHardwareInfo(ResultListener listener);

返回结果：

参看：com.aiedevice.sdkdemo.bean.HardwareInfoData

7. 修改音量

7.1. 功能说明

设备端音量值有效值是 0-100，但是不是每个值都是合理值。设备端一共有 7 档音量，在相同区间范围的数值音量大小是相同的，7 档区间分别是：

{0-40, 41-50, 51-60, 61-80, 81-84, 85-92, 93-100};

7.2. 接口说明

1) 获取设备信息

```
// targetVolume: 修改后的音量值（必选）
public void changeDeviceVolume(int targetVolume, ResultListener listener);
```

返回结果：

参看：com.aiedevice.bean.Result

```
    "result": 0,
    "msg": "success"
}
```

失败：

```
{
    "result": errCode,
    "msg": errMessage
}
```