



# 快速集成安卓 App SDK

文档版本: 20251114

[查看在线版本](#)

## 目录

<b>1. 前提条件 .....</b>	<b>2</b>
<b>2. SDK 版本 .....</b>	<b>3</b>
<b>3. 集成 SDK .....</b>	<b>4</b>
3.1. 第一步：创建 Android 工程 .....	4
3.2. 第二步：配置 build.gradle 文件 .....	4
3.3. 第三步：集成安全组件 .....	5
3.4. 第四步：设置 Appkey、AppSecret 和证书签名 .....	5
3.5. 第五步：混淆配置 .....	6
3.6. 第六步：初始化 SDK .....	7
3.7. 第七步：注销云连接 .....	7
3.8. 第八步：开启或关闭日志 .....	7
<b>4. 运行 Demo 应用 .....</b>	<b>9</b>
4.1. Demo 应用介绍 .....	9
4.2. 运行 Demo .....	11
<b>5. 常见问题 .....</b>	<b>12</b>
5.1. SING_VALIDATE_FALED .....	12
5.2. ILLEGAL_CLIENT_ID .....	12

本文介绍如何将安卓版涂鸦 智能生活 App SDK 集成到您的开发环境中，例如 Android Studio，并介绍初始化方法以及如何启用调试模式。然后，您可以尝试运行 Demo，快速上手全屋智能移动应用开发。

## 1. 前提条件

- 开始操作前, 请确保您已经完成 [准备工作](#)。
- 如果您还未安装 **Android Studio**, 请访问 [安卓官网](#) 进行下载安装。

## 2. SDK 版本

- 如果您之前采用的是从 4.x.x 或者 3.x.x 版本智能生活 App SDK，请参考 [迁移指南](#) 升级到最新版本 SDK。
- 从 4.0.0 版本智能生活 App SDK 开始，SDK 分为开发版和正式版。详情请参考 [产品定价](#)。
- 从 4.0.0 版本智能生活 App SDK 开始，在涂鸦开发者平台配置的安卓包名需要与您的工程 `packageName` 保持一致。否则，系统会报错 `ILLEGAL_CLIENT_ID`。
- 从 3.29.5 版本智能生活 App SDK 开始，您需要为 SDK 应用设置 SHA256 密钥。详情请参考 [如何获取 SHA 密钥](#)。

如果您需要上传应用至 Google Play 应用商店，请注意查看是否已开启重签名功能。若已开启，需要将谷歌生成的 SHA256 密钥配置到涂鸦开发者平台，否则会提示 **非法客户端** 问题。详情请参考 [开启 Android 签名保护](#)。

- 3.10.0 及之前版本的 SDK 只支持安卓 armeabi-v7a。3.11.0 版本后已经将 armeabi-v7a、arm64-v8a 集成进 SDK。您需要将本地自行放入的 SDK 的相关 so 库移除，使用 SDK 中提供的。如果集成新版本 so 库，请移除之前老版本手动集成的库，防止冲突或者代码版本不一致导致的问题。

## 3. 集成 SDK

### 3.1. 第一步：创建 Android 工程

在 Android Studio 中新建工程。

### 3.2. 第二步：配置 build.gradle 文件

在安卓项目的 `build.gradle` 文件里，添加集成准备中下载的 `dependencies` 依赖库。

```
1 android {
2     defaultConfig {
3         ndk {
4             abiFilters "armeabi-v7a", "arm64-v8a"
5         }
6     }
7     packagingOptions {
8         pickFirst 'lib/*/libc++_shared.so' // 多个 AAR (Android Library) 文件中存在
9         此 .so 文件，请选择第一个
10    }
11 }
12 configurations.all {
13     exclude group: "com.thingclips.smart" ,module: 'thingsmart-modularCampAnno'
14 }
15
16 dependencies {
17     implementation fileTree(dir: 'libs', include: ['*.aar'])
18     implementation 'com.alibaba:fastjson:1.1.67.android'
19     implementation 'com.squareup.okhttp3:okhttp-urlconnection:3.14.9'
20
21     // App SDK 最新稳定安卓版:
22     implementation 'com.thingclips.smart:thingsmart:6.11.1'
23 }
```

在根目录的 `build.gradle` 文件中，增加涂鸦 Maven 仓库地址，进行仓库配置。

```
1 repositories {
2     jcenter()
3     maven { url 'https://maven-other.tuya.com/repository/maven-releases/' }
4     maven { url "https://maven-other.tuya.com/repository/maven-commercial-
5     releases/" }
6     maven { url 'https://jitpack.io' }
7     google()
8     mavenCentral()
9     maven { url 'https://maven.aliyun.com/repository/public' }
10    maven { url 'https://central.maven.org/maven2/' }
11    maven { url 'https://oss.sonatype.org/content/repositories/snapshots/' }
12 }
```



该 SDK 版本支持的安卓 minSdkVersion 为 23， targetSdkVersion 为 35，且仅支持通过 AndroidX 构建。

### 3.3. 第三步：集成安全组件

- 在 [获取 SDK](#) 页面，勾选一款或多款您需要的 SDK 或者业务包，然后下载对应应用平台的集成资料包。

[苹果 下载 iOS 开发版](#)[安卓 下载 Android 开发版](#)[快速集成iOS SDK](#)[快速集成Android SDK](#)

- 将下载后的资料包解压，并将 security-algorithm.aar 放置到工程 libs 目录下，并确认工程 build.gradle 的 dependencies 中有以下依赖： implementation

```
fileTree(include: ['*.aar'], dir: 'libs')
```

```
dependencies {  
    implementation fileTree(dir: 'libs', include: ['*.aar'])  
}
```

### 3.4. 第四步：设置 Appkey、AppSecret 和证书签名

- 在 [涂鸦开发者平台](#)，找到您创建的 SDK。

The screenshot shows the Tuya developer console interface. At the top, there are three tabs: '获取SDK' (Get SDK), '获取密钥' (Get Key), and '独立域名服务' (Independent Domain Name Service). The '获取密钥' tab is selected. Below it, there are two sections: 'iOS' and 'Android'. Each section has a logo, a title ('iOS密钥' or 'Android密钥'), and two text fields: 'AppKey' and 'AppSecret'. Each text field has a placeholder with a series of asterisks and a '显示' (Show) link. Below each text field is a '复制' (Copy) button.

**iOS**

AppKey: 显示 复制

AppSecret: 显示 复制

**Android**

Android密钥

AppKey: 显示 复制

AppSecret: 显示 复制

**证书**

设置 SHA256 可提升 App 的安全性。生成 SHA256 方式可查看 [说明文档](#)。每天最多新增5个，一共最多可以设置10个 SHA256。

新增SHA256:

如您的App未配置过SHA-256重签名，可不填；保存后无...

2. 在上图中，获取 AppKey，以及 AppSecret，并配置在 `AndroidManifest.xml` 中：

```

1 <meta-data
2   android:name="THING_SMART_APPKEY"
3   android:value="应用 Appkey" />
4 <meta-data
5   android:name="THING_SMART_SECRET"
6   android:value="应用密钥 AppSecret" />
```

3. 配置应用证书：

1. 生成一个 SHA256 密钥。应用证书请参考 [Android 官方文档](#)，以及 [如何获取证书 SHA256](#)。
2. 将您的 SHA256 密钥填入的 **证书** 中。

### 3.5. 第五步：混淆配置

在 `proguard-rules.pro` 文件配置相应混淆配置。

```

1 #fastJson
2 -keep class com.alibaba.fastjson.**{*;}
3 -dontwarn com.alibaba.fastjson.**
4
5 #mqtt
6 -keep class com.thingclips.smart.mqttclient.mqttv3.** { *; }
7 -dontwarn com.thingclips.smart.mqttclient.mqttv3.**
8
9 #OkHttp3
```

```
10 -keep class okhttp3.** { *; }
11 -keep interface okhttp3.** { *; }
12 -dontwarn okhttp.**
13
14 -keep class okio.** { *; }
15 -dontwarn okio.**
16
17 -keep class com.thingclips.**{*;}
18 -dontwarn com.thingclips.**
19
20 # Matter SDK
21 -keep class chip.** { *; }
22 -dontwarn chip.**
23
24 #MINI SDK
25 -keep class com.gzl.smart.** { *; }
26 -dontwarn com.gzl.smart.**
```

### 3.6. 第六步：初始化 SDK

您需要在 Application 的主线程中初始化 SDK，确保所有进程都能初始化。示例代码如下：

```
1 public class ThingSmartApp extends Application {
2     @Override
3     public void onCreate() {
4         super.onCreate();
5         ThingHomeSdk.init(this);
6     }
7 }
```



appKey 和 appSecret 可以配置在 AndroidManifest.xml 文件里，也可以在初始化代码里初始化。

```
1 ThingHomeSdk.init(Application application, String appkey, String appSerect)
```

### 3.7. 第七步：注销云连接

在退出应用的时候，调用以下接口可以注销云连接。

```
1 ThingHomeSdk.onDestroy();
```

### 3.8. 第八步：开启或关闭日志

- 在 **debug** 模式下，您可以开启 SDK 的日志开关，查看更多的日志信息，帮助您快速定位问题。

- 在 **release** 模式下，建议关闭日志开关。

```
1 ThingHomeSdk.setDebugMode(true);
```

## 4. 运行 Demo 应用



- 在完成快速集成 SDK 后，您将获取到 SDK 使用的 AppKey、AppSecret 信息。集成 SDK 时，请确认 AppKey、AppSecret 是否与平台上的信息一致，任意一个不匹配会导致 SDK 无法使用。详细操作，请参考 [第四步：集成和设置 Appkey 和 AppSecret](#)。
- 该智能生活 App SDK 示例工程 Demo 仅用于演示体验，请勿直接商用。更多信息，请访问 [《涂鸦开发服务协议》](#)。

Demo 应用演示 App SDK 的开发流程，展现如何调用 SDK 能力、实现智能家居场景。在开发应用之前，建议您根据需要先按照以下流程完成 Demo 应用的操作。

### 4.1. Demo 应用介绍

Demo 应用主要包括：

- 用户管理：使用手机号或者邮箱进行登录和注册。
- 家庭管理和设备管理：
- 创建家庭，切换用户所属的当前家庭。
- 展现家庭中设备列表，控制设备功能。
- 设备重命名和设备移除。
- 设备配网：包括 Wi-Fi 快连配网模式、热点配网模式、有线网关配网、网关子设备配网、蓝牙配网、Mesh 子设备配网。

## 涂鸦智能

用户管理

用户信息 >

退出

家庭管理

添加家庭 >

当前家庭 >

家庭列表 >

设备配网

EZ 模式 >

AP 模式 >

蓝牙 BLE 模式 >

Zigbee 网关 >

Zigbee 子设备 >

QR 扫码 >

Mesh 子设备 >

设备管理

设备列表 >

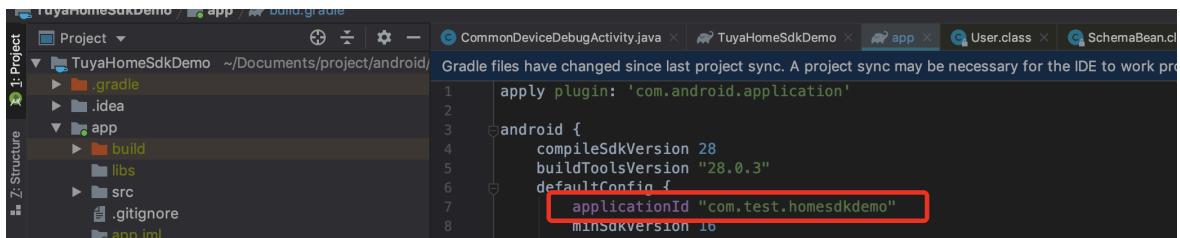
Zigbee 网关列表 >

群组列表 >

更多详情, 请访问 [tuya-home-android-sdk-sample-kotlin GitHub 项目](#)。

## 4.2. 运行 Demo

1. 替换 app 目录下 build.gradle 文件中的 applicationId 为您的应用包名。



```
apply plugin: 'com.android.application'
android {
    compileSdkVersion 28
    buildToolsVersion "28.0.3"
    defaultConfig {
        applicationId "com.test.homesdkdemo"
        minSdkVersion 16
    }
}
```

2. 确认您已经完成 [第三步：集成安全组件](#) 以及 [第四步：设置 Appkey, AppSecret 和证书签名](#)。
3. 然后单击运行，运行 Demo。

## 5. 常见问题

### 5.1. SING\_VALIDATE\_FALED

API 请求提示签名错误（`SING_VALIDATE_FALED`）

- 问题现象：运行 Demo 时提示以下错误：

```
1  {
2      "success": false,
3      "errorCode" : "SING_VALIDATE_FALED",
4      "status" : "error",
5      "errorMsg" : "Permission Verification Failed",
6      "t" : 1583208740059
7 }
```

- 解决方法：

请检查您的 AppKey、AppSecret 是否正确配置，是否和 [准备工作](#) 中获取到的一致。

### 5.2. ILLEGAL\_CLIENT\_ID

- 为什么智能生活 App SDK 升级到 3.29.5 版本后，报错 `ILLEGAL_CLIENT_ID`？  
因为智能生活 App SDK 从 3.29.5 版本开始，做了安全校验的升级。  
您需要在涂鸦开发者平台根据说明文档来 [获取 SHA256](#)，然后在平台绑定您的 SHA256。  
详细配置步骤，请参考 [准备工作](#)。
- 为什么智能生活 App SDK 升级到 4.0.0 后，报错 `ILLEGAL_CLIENT_ID`？  
因为智能生活 App SDK 从 4.0.0 版本开始，做了安全校验的升级。  
您在涂鸦开发者平台配置的安卓包名需要与您的工程 `packageName` 保持一致。否则，系统会报错 `ILLEGAL_CLIENT_ID`。
- 为什么在涂鸦开发者平台配置了 `SHA256`，直接运行 Demo，还是报错 `ILLEGAL_CLIENT_ID`？

解决方法：直接运行 Demo 前，需要在 `app` 模块的 `build.gradle` 中配置您自己的签名信息：

```
1 android {
2     ...
3     signingConfigs {
4         debug {
5             storeFile file('xxxx.jks')
6             storePassword 'xxxxxx'
7             keyAlias 'xxx'
8             keyPassword 'xxxxxx'
9         }
10        release {
```

```
11     storeFile file('xxxx.jks')
12     storePassword 'xxxxxx'
13     keyAlias 'xxx'
14     keyPassword 'xxxxxx'
15   }
16 }
17 }
```