畅联科技锁掌柜开放接口文档 — Android 版本

用户手册

广州畅联信息科技有限公司

2019-09-04

一、引入到项目

1、Android Studio 引入

配置 Android manifest.xml，添加权限：

```
<uses-permission android:name="android.permission.BLUETOOTH"/>
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
<uses-permission android:name="android.permission.ACCESS_WIFI_STATE"/>
```

在 build.gradle 里面添加依赖：

```
allprojects {
    repositories {
        ...
        maven { url 'https://jitpack.io' }
    }
}

dependencies {
  ...
   // okhttp
  implementation 'com.squareup.okhttp3:okhttp:3.8.0'
  // retrofit
  implementation 'com.squareup.retrofit2:retrofit:2.3.0'
  implementation 'com.google.code.gson:gson:2.8.5'
  implementation 'com.squareup.retrofit2:converter-gson:2.3.0'
  implementation 'com.squareup.retrofit2:adapter-rxjava:2.3.0'
  //rx
  implementation 'io.reactivex:rxandroid:1.2.1'
  implementation 'io.reactivex:rxjava:1.3.0'
  // glide
  implementation 'com.github.bumptech.glide:glide:4.7.1'
  annotationProcessor 'com.github.bumptech.glide:compiler:4.7.1'
  // rxpermission
  implementation 'com.github.tbruyelle:rxpermissions:0.10.2'
  //autoLayout
  implementation 'com.zhy:autolayout:1.4.5'
  implementation 'com.uclbrt:QRMasterBTSDK:0.9.8'
```

```
}
```

2、Eclipse 引入

暂缺教程

二、配置开始使用

1、分配钥匙

1.1 实例化生成类

```
mKeyGenerator = new KeyGenerator(Config.UCLBRT_SID, Config.UCLBRT_TOKEN);
```

1.2 配置集群号和请求回调

```
mKeyGenerator.setListener(new KeyGeneratorListener() {
  @Override public void onSuccess(String cardNo) {
    // TODO 后台线程返回结果
    // TODO 运行在 main 线程
    hideLoading();
    mCardNo = cardNo;
    ToastUtils.show(MainActivity.this, "生成房卡 " + cardNo);
  }

  @Override public void onError(String message, int status) {
    // TODO 后台线程返回结果
    // TODO 运行在 main 线程
    hideLoading();
    ToastUtils.show(MainActivity.this, message);
  }
});
```

1.3 分配某房间钥匙给某手机号，发起请求

```
mKeyGenerator.createCard(mobile, communityNo, buildNo, floorNo, roomNo,
startTimeStr,endTimeStr, 86);
```

2、获取钥匙

2.1 实例化获取钥匙 Fragment

2.2 配置开发者 id，token，集群编号和请求回调

```
@Override protected void onCreate(Bundle savedInstanceState) {
  super.onCreate(savedInstanceState);
// 获取钥匙
keyFetcher = new KeyFetcher(this, Config.UCLBRT_SID, Config.UCLBRT_TOKEN);
```

```
// 必须
keyFetcher.onCreate(this);
}

@Override protected void onDestroy() {
  super.onDestroy();
  // 必须
  keyFetcher.onDestroy(this);
}
```

2.3 获取钥匙，发起请求

获取钥匙前，需要先调用 createCard 创建房卡。

```
// 通过楼栋房间信息获取钥匙
// 通过 getCardByRoom 获取钥匙，须与 createCard 的参数一致。
keyFetcher.getCardByRoom(communityNo, buildNo, floorNo, roomNo, startTimeStr,
endTimeStr,"86", mobile);

// 使用 CardNo 来获取钥匙
// 传入 createCard 获取的 CardNo
keyFetcher.getCardByCardNo(communityNo, mCardNo, "86", mobile);
```

2.4 钥匙获取监听

```
// 监听获取钥匙回调
keyFetcher.setKeyFetcherListener(new KeyFetcherListener() {
  @Override public void onSuccess(Fragment fragment) {
    hideLoading();

    // 需要先判断 Fragment 是否为空
    if (fragment != null) {
      // 加载 qrb 开门界面
      FragmentManager manager = getSupportFragmentManager();
      FragmentTransaction transaction = manager.beginTransaction();
      transaction.replace(R.id.content, fragment);
      transaction.show(fragment);
      // 可选 添加获取钥匙错误显示的 Fragment
      transaction.commit();
    } else {
      showLoading();
      // 共享版，没有界面，直接连接蓝牙开门
    }
  }

  @Override public void onError(String message, int status) {
```

```
    ToastUtils.show(MainActivity.this, message);
    hideLoading();
  }
});
```

2.5 添加蓝牙开门监听

```
// 设置开门监听
  keyFetcher.setOpenDoorListener(new IOpenDoorListener() {
    @Override public void onOpenDoorSuccess(String msg) {
      // TODO 后台线程返回结果
      // TODO 运行在 main 线程
      hideLoading();
      ToastUtils.show(MainActivity.this, msg);
    }

    @Override public void onOpenDoorFailure(int errorCode) {
      // TODO 后台线程返回结果
      // TODO 运行在 main 线程
      hideLoading();
      ToastUtils.show(MainActivity.this,
    UclbrtBleLockErrorCode.getErrorStr(MainActivity.this, errorCode));
    }
  });
}
```

2.6 IOpenDoorListener 监听开门失败返回的 errorCode 对照表

```
public class UclbrtBleLockErrorCode {
  public static final int UNKNOWN_ERROR = -1;
  public static final int SUCCESS = 0x01;//操作成功
  public static final int KEY_ERROR = 0x03;//密钥不对
  public static final int CARD_TYPE_ERROR = 0x04;//卡类型不正常
  public static final int NO_CUSTOMER_ERROR = 0x05;//没入住客户
  public static final int ROOM_NUMBER_ERROR = 0x06;//房间号不对
  public static final int TIME_ERROR = 0x07;//时间不对
  public static final int BATCH_ERROR = 0x08;//批次号不对
  public static final int ROOM_CLOSED_ERROR = 0x0D;//房间反锁
  public static final int HOTEL_NUMBER_ERROR = 0x0F;//酒店标示不对
  public static final int AGENTS_NUMBER_ERROR = 0x1E;//代理商号不对
  public static final int LOW_BATTERY_ERROR = 0x0E; // 电池电量过低
  public static final int INEFFECTIVE_TIME_ERROR = 0x12; // qra 门锁有效次数无
效
  public static final int INVALID_BRAND_ERROR = 0x14; // 品牌商不对
```

```
  public static final int BLE_NOT_OPEN_ERROR = -2;  // 没有开启蓝牙
  public static final int BLE_PERMISSION_DENIED_ERROR = -3;   // 没有蓝牙权限
  public static final int BLE_CONNECT_ERROR = -4;   // 蓝牙连接失败
}
```

使用
UclbrtBleLockErrorCode.*getErrorStr*(Context, errorCode)
解释错误码的释义

3、让钥匙失效

3.1 实例化失效类

```
// 失效房卡类
mKeyInvalid = new KeyInvalid(Config.UCLBRT_SID, Config.UCLBRT_TOKEN);
```

3.2 配置集群编号和请求回调

```
mKeyInvalid.setCommunityNo(Config.UCLBRT_COMMUNITY_NO);
mKeyInvalid.setListener(new KeyInvalidListener() {
  @Override public void onSuccess() {
    // TODO 后台线程返回结果
    // TODO 运行在 main 线程
    hideLoading();
    ToastUtils.show(MainActivity.this, "房卡已失效");
  }

  @Override public void onError(String message, int status) {
    // TODO 后台线程返回结果
    // TODO 运行在 main 线程
    hideLoading();
  }
});
```

3.3 让某手机号被分配在某房间的钥匙失效，发起请求

```
mKeyInvalid.cancelCard(mCardNo);
```

4、传入钥匙串进行开门

4.1 设置开门监听

```
//使用此方法开门，id，token 可以设置为空
  keyFetcher = new KeyFetcher(this, "", "");
// 设置开门监听
  keyFetcher.setOpenDoorListener(new IOpenDoorListener() {
    @Override public void onOpenDoorSuccess(String msg) {
```

```
        // TODO 后台线程返回结果
        // TODO 运行在 main 线程
        hideLoading();
        ToastUtils.show(MainActivity.this, msg);
      }

      @Override public void onOpenDoorFailure(String errorMsg) {
        // TODO 后台线程返回结果
        // TODO 运行在 main 线程
        hideLoading();
        ToastUtils.show(MainActivity.this, errorMsg);
      }
    });
}
```

4.2 传入钥匙串进行蓝牙开门

```
try {
  keyFetcher.openDoor(this, bleStr);
} catch (BleStrException e) {
    ToastUtils.show(this, "输入的 bleStr 格式不正确");
}
```

5、超时设置（此版本新增）

5.1 设置获取获取钥匙等网络连接的超时时间

在你的 Application 中

```
public class SampleApplication extends Application {

  @Override public void onCreate() {
    super.onCreate();

    // 设置获取获取钥匙等网络连接的超时时间，单位毫秒，默认为 10000 毫秒
    BTSDKConfig.getInstance().setHttpConnectTimeout(10000);
    BTSDKConfig.getInstance().setHttpReadTimeout(10000);
    BTSDKConfig.getInstance().setHttpWriteTimeout(10000);

    // other initial...
  }
}
```

5.2 设置蓝牙扫描、蓝牙发送数据的超时

```
// 设置蓝牙的连接超时时间，单位毫秒，默认 30000 毫秒
keyFetcher.setBleScanTimeout(30000);
```

```
// 设置蓝牙发送数据的超时时间，单位毫秒，默认 10000 毫秒
keyFetcher.setBleSendDataTimeout(10000);
```

三、Proguard 混淆

```
# okhttp
-dontwarn com.squareup.okhttp3.**
-keep class com.squareup.okhttp3.** { *;}
-dontwarn okio.*

# retrofit
-dontwarn retrofit2.**
-keep class retrofit2.** { *; }
-keepattributes Signature
-keepattributes Exceptions
# Retain generic type information for use by reflection by converters and adapters.
-keepattributes Signature
# Retain service method parameters when optimizing.
-keepclassmembers,allowshrinking,allowobfuscation interface * {
    @retrofit2.http.* <methods>;
}
# Ignore annotation used for build tooling.
-dontwarn org.codehaus.mojo.animal_sniffer.IgnoreJRERequirement
# Ignore JSR 305 annotations for embedding nullability information.
-dontwarn javax.annotation.**


# Gson
-keep class com.google.gson.stream.** { *; }
-keepattributes EnclosingMethod
-keep class com.google.gson.** {*;}
-keep class com.google.**{*;}
-keep class sun.misc.Unsafe { *; }
-keep class com.google.gson.examples.android.model.** { *; }

# glide
-keep public class * implements com.bumptech.glide.module.GlideModule
-keep public class * extends com.bumptech.glide.module.AppGlideModule
-keep public enum com.bumptech.glide.load.ImageHeaderParser$** {
```

```
  **[] $VALUES;
  public *;
}
# RxJava RxAndroid
-dontwarn sun.misc.**
-keepclassmembers class rx.internal.util.unsafe.*ArrayQueue*Field* {
    long producerIndex;
    long consumerIndex;
}
-keepclassmembers class rx.internal.util.unsafe.BaseLinkedQueueProducerNodeRef {
    rx.internal.util.atomic.LinkedQueueNode producerNode;
}
-keepclassmembers class rx.internal.util.unsafe.BaseLinkedQueueConsumerNodeRef {
    rx.internal.util.atomic.LinkedQueueNode consumerNode;
}
```