Android SDK集成文档 v1.1

目录

一. SDK jar包说明

- **ROBOTPENSDK**
  核心服务，用于与笔建立连接、通信。同时提供录制模块和一些常用工具。

- **ROBOTPENMODEL**
  SDK使用到的一些对象。

- **ROBOTPENFILE**
  文件服务库，用于访问云盘资源。

- **ROBOTPENREMOTE**
  远端服务库，用于白板交互。

二. 项目集成SDK

将SDK所有jar包导入到项目中。

三. 配置AndroidManifest.xml

- **添加SDK需要的权限**

```
<!-- 访问网络权限 -->
<uses-permission android:name="android.permission.INTERNET" />
<!-- 获取音频权限 -->
<uses-permission android:name="android.permission.RECORD_AUDIO" />
<!-- 写入存储权限 -->
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
 <uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
 <uses-permission android:name="android.permission.MOUNT_UNMOUNT_FILESYSTEMS" />
```

- **添加需要的服务**

```
<!-- USB笔服务 -->
<service android:name=“cn.robotpen.core.services.UsbPenService" android:enabled="true" />
<!-- 文件管理服务 -->
<service android:name=“cn.robotpen.file.services.FileManageService" android:enabled="true" /
>
```

四. 笔交互基础服务

- **绑定笔服务**
  1. 项目Application继承PenApplication类；
  2. 在启动APP时调用mApplication.bindPenService(mApplication.getConnectDeviceType()) 绑定服务，退出APP时调用mApplication.unBindPenService() 解绑服务；
  3. 程序启动时，确保mApplication.getPenService()返回的mPenService不为NULL；

4. 设置用户标识 mPenService.setUserId(userId)；
5. 设置设备类型为10.1寸 mPenService.setSceneType(SceneType.INCH_101)，如果需要横屏显示那么设置为SceneType.INCH_101_horizontal。

- **监听设备连接状态**

  设置连接状态监听：
  mPenService.setOnConnectStateListener(Listeners.OnConnectStateListener listener);

  通过OnConnectStateListener.stateChange可以获得连接状态：
  CONNECTING：连接中；
  CONNECTED：连接完成；
  DISCONNECTED：设备已断开。

  以上状态当设备插入或拔出时会主动触发。

- **主动查找设备**

  调用mPenService.scanDevice(null) 触发服务主动查找设备；
  查找状态会通过OnConnectStateListener 返回。

- **获取设备坐标**

  设置笔坐标监听：
  mPenService.setOnPointChangeListener(OnPointChangeListener listener);

  当笔连接成功后，会通过OnPointChangeListener返回PointObject对象。
  1. 设置PointObject的showUIWidth和showUIHeight值为当前UI画布显示区域的值；
  2. 通过PointObject的getSceneX()和getSceneY()可获得笔坐标缩放到当前UI画布后的坐标。

五. 白板模块

在RobotpenSDK中，提供了白板模块MultipleCanvasView，实现了背景颜色、背景图片、笔迹颜色、笔迹粗细、橡皮擦、插入图片和清屏等功能。

- **使用方法**

  MultipleCanvasView view = new MultipleCanvasView(Context context,CanvasManageInterface canvasManage);
  addView(view);

- **实现CANVASMANAGEINTERFACE接口**

  CanvasManageInterface接口用于让白板获取当前需要显示的背景颜色、背景图片、笔迹颜色、笔迹粗细和是否是橡皮擦状态。

```
/**
 * 获取笔模式
 * @return
 */
```

```
PenModel getPenModel();

/**
 * 获取笔的粗细,建议以2为起始值
 * @return
 */
float getPenWeight();

/**
 * 获取笔颜色
 * @return
 */
int getPenColor();

/**
 * 获取背景颜色
 * @return
 */
int getBgColor();

/**
 * 获取背景图片
 * @return
 */
int getBgResId();

/**
 * 获取尺寸对象
 * @return
 */
FrameSizeObject getFrameSize();

/**
 * 获取是否是橡皮擦状态
 * @return
 */
boolean getIsRubber();

/**
 * 输出绘画状态
 * @param isRoute
 */
void penRouteStatus(boolean isRoute);

/**
 * 获取轨迹监听
 * @return
 */
TrailsManageModule.OnTrailsListener getTrailsListener();

/**
 * 获取当前用户
 * @return
 */
```

```
int getCurrUserId();

/**
*获取背景图片
*@return
*/
public Bitmap getBgBitmap()
/**
*获取背景图片的缩放形式
*@return
*/
public ScaleType getBgScaleType()
```

六. 白板录课功能

在RobotpenSDK中，提供了录制模块ImageRecordModule，使用它可以轻松的将View录制成MP4视频。

- **使用方法**

首先需要加载SDK包jniLibs目录里的JNI视频库，然后在程序中加载，建议放在Application里

```
static {
    System.loadLibrary("avutil-54");
    System.loadLibrary("swresample-1");
    System.loadLibrary("swscale-3");
    System.loadLibrary("postproc-53");
    System.loadLibrary("avcodec-56");
    System.loadLibrary("avformat-56");
    System.loadLibrary("avfilter-5");
    System.loadLibrary("RecordImageUtil");
}

//初始化录制工具
mImageRecordModule = new ImageRecordModule(this);
mImageRecordModule.setSavePhotoDir(设置保存图片的路径);
mImageRecordModule.setSaveVideoDir(设置保存视频的路径);
mImageRecordModule.setRecordLevel(RecordLevel level);      //设置录制等级
mImageRecordModule.init();

//保存View截图
mImageRecordModule.saveSnapshot()

//是否正在录制
mImageRecordModule.getIsRecording()


/**
 * 设置录制尺寸信息
 * @param inImageWidth          输入图片宽
 * @param inImageHeight         输入图片高
 * @param outVideoWidth         输出视频宽
```

```
     * @param outVideoHeight          输出视频高
     * @return
     */
    public boolean setRecordSize(int inImageWidth,int inImageHeight,int outVideoWidth,int
outVideoHeight)


    /**
     * 开始录制
     */
    public boolean startRecord()


    /**
     * 结束录制<br />
     * 结束后并不表示录制完全结束，需要等待ImageRecordInterface.videoCodeState返回的
progress=100后才表示完全压缩完成。<br />
     * 如果录制完成后需等待压缩时间很长，那么可能导致音视频时间不同步，需降低RecordLevel
参数
     */
    public void endRecord()


    /**设置暂停**/
    public void setIsPause(boolean value)
```

- **实现IMAGERECORDINTERFACE接口**

```
    /**
     * 录制时，通过该接口获取view截图缓存
     * @param buffer
     * @return 返回读取长度
     */
    int fillImageBuffer(ByteBuffer buffer);

    /**
     * 当前录制的时间
     * @param second
     */
    void recordTimeChange(int second);

    /**
     * 如果结束后视频压缩还未完成，那么返回压缩进度
     * @param progress
     */
    void videoCodeState(int progress);

    /**
     * 如果录制过程中出现问题，如内存不足，会通过该接口发出警告
     * @param state
     */
    void recordWarning(RecordState state);
```

- **RECORDLEVEL录制等级**

常用等级：

```
/**标清320p 10fps**/
public static final int level_3 = 3;

/**高清480p 10fps**/
public static final int level_13 = 13;

/**超清720p 10fps**/
public static final int level_23 = 23;
```

七. 白板P2P交互

首先设置MultipleCanvasView.CanvasManageInterface的getTrailsListener监听，获取白板的轨迹信息，通过mPenService发送出去：

```
@Override
public TrailsManageModule.OnTrailsListener getTrailsListener() {
    return new TrailsManageModule.OnTrailsListener(){
        @Override
        public void sendTrails(String trailsJson) {
                mPenService.sendTrails(trailsJson);
        }
    };
}
```

然后需要设置mPenService的OnTrailsClientChangeListener轨迹监听，接收对方传来的轨迹；

```
/**
 * 设置轨迹监听，用于远程交互
 * @param listener
 */
public void setOnTrailsClientChangeListener(Listeners.OnTrailsClientChangeListener listener)
```

最后使用mPenService提供的方法：

```
/**
 * 设置当前用户ID
 * @param value
 */
public void setUserId(int value)

/**
 * 设置P2P对方用户ID，优先级大于GroupId，如果要取消P2P那么设置为0即可
 * @param value
 */
public void setLiveTargetId(int value)
```

```
/**
 * 设置直播组ID
 * @param value
 */
public void setmLiveGroupId(String value)

/**
 * 开始直播
 */
public void startLive()

/**
 * 结束直播
 */
public void stopLive()
```

完成以上操作就能在MultipleCanvasView中实现远程P2P交互了。