VCS SDK 使用说明文档

1 依赖配置

拿到**VCS**的**SDK**：**swap-xxx.aar** , 放到集成模块的**libs**目录下， 并在模块的**build.gradle**中增加依赖

```
dependencies {
    ......
    implementation files('libs/swap-xxx.aar')
    ......
}
```

2 swap SDK 使用说明

新建Service, 并在 AndroidManifest.xml中注册

注册Service类

```
<application>
    ......
        <service
            android:name=".XxxService"
            android:enabled="true"
            android:exported="true" />
    ......
</application>
```

新建Service类

```
public class XxxService extends Service {
    private static final String TAG = "XxxService";

    //vcbNotificationManager 用于主动通知VCS事件
    private VcbNotificationManager mVcbNotificationManager;

    //mBridgeListeners 用于保存客户端注册的监听器，用于异步返回给客户端操作结果
    private final Set<IBridgeListener> mBridgeListeners = new HashSet<>();

    // 实现aidl stub
    private final IBridge.Stub mBind = new IBridge.Stub() {
        @Override
        public String sendRequest(String uuid, String msg) throws RemoteException {
            //接收VCS发来的json指令。
            // 可以同步返回结果：组装成json返回；
            // 不能同步处理结果：返回null，然后通过listener通知处理结果；
            xxxFunction(uuid);
            return null;
        }

        @Override
        public void registerBridgeListener(IBridgeListener bridgeListener) throws RemoteException {
            mBridgeListeners.add(bridgeListener);
        }
    };

    private void xxxFunction(final String uuid) throws RemoteException {
        //处理数据，调用接口，返回结果或通过listener回调给VCS
        String callbackMsg = "";

        for (IBridgeListener listener : mBridgeListeners) {
            if (null != listener) {
                // 操作结果调用 onResponse
                listener.onResponse(uuid, callbackMsg);
            }
        }
        // 这里如果需要播报TTS，或其他事件通知到VCS，调用下面方法
        String notifyUuid = UUID.randomUUID().toString();
        String notifyEvent = ""; //组装好的json字符串，格式待定义
        if (null != mVcbNotificationManager){
            mVcbNotificationManager.notifyEvent(notifyUuid, notifyEvent);
        }
    }

    @Override
    public void onCreate() {
        super.onCreate();
        mVcbNotificationManager = new VcbNotificationManager(this);
    }

    @Override
    public IBinder onBind(Intent intent) {
```

```
        return mBind;
    }

    @Override
    public boolean onUnbind(Intent intent) {
        return super.onUnbind(intent);
    }

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

3 swap SDK 接口说明

3.1 IBridge.Stub 接口说明

接口名称：客户端发送请求控制、查询...

```
String sendRequest(String uuid, String msg)
```

接口参数

`uuid`： 标识此次请求的唯一识别码

`msg`： json格式指令字符串

返回值：

null 或 json 格式字符串

> null表示此次请求无法同步返回给客户端，需要客户端通过监听器接收指令执行结果
> json格式字符串表示此次请求的请求结果，例如：
>
> ```
> {
>   "code":0,
>   "msg":"操作成功"
> }
> ```
>
> 具体格式根据不同请求返回不同格式

3.2 IBridgeListener 接口说明

3.2.1 接口名称：服务端指令执行返回响应

```
void onResponse(String uuid, String msg)
```

接口参数

`uuid`： 返回码，需要和请求码一致

`msg`： 额外信息，可以为null、string、json格式字符串

3.3 VcbNotificationManager 接口说明

3.3.1 接口名称：服务端主动通知事件接口

```
void notifyEvent(@NonNull String uuid, String event)
```

接口参数

`uuid`： 事件唯一标识码

`event`： 事件信息，json格式

3.4 TtsClientManager 接口说明

3.4.1 接口名称：构造函数

```
TtsClientManager(@NonNull Context context)
```

接口参数

`context`： 上下文

3.4.2 接口名称：构造函数

```
TtsClientManager(@NonNull Context context, @NonNull String packageName, @NonNull String serviceName)
```

接口参数

`context`： 上下文

`packageName`： VCS 包名

`serviceName`： VCS 中TTS服务名

3.4.3 接口名称：合成播报文本

```
int startTts(String text, TtsOption option)
```

接口参数

`text`： 待合成的文本

`option`： 合成文本时的选项

3.4.4 接口名称：合成播报文本并监听

```
int startTts(String text, TtsOption option, ITtsListener listener)
```

接口参数

`text`： 待合成的文本

`option`： 合成文本时的选项

`listener`： 监听器

3.4.5 接口名称：设置播报监听器

```
void setTtsListener(ITtsListener ttsListener)
```

接口参数

`listener`： 监听器

3.5 TtsOption

- `role`：发音人角色
- `speed`：语速
- `pitch`：语调
- `level`：等级
- `nluType`：附带反馈类型（用于播放TTS的同时通知APA模块）
- `audioFocusType`：音频焦点类型
- `background`：前后台播报（前台：向用户展示播放的文本；后台：只播放声音，不展示文本）

3.6 ITtsListener

3.6.1 接口名称：开始播放监听

```
void onSpeechStart()
```

3.6.2 接口名称：播放完成监听

```
void onSpeechFinished()
```

3.6.3 接口名称：播放被打断监听

```
void onSpeechInterrupted()
```

3.6.4 接口名称：播放出错监听

```
void onSpeechError(int errorCode)
```

接口参数

`errorCode` ： 错误码

3.7 VoiceSettingManager

3.7.1 接口名称：语音设置管理器构造函数

```
VoiceSettingManager(@NonNull Context context)
```

接口参数

`context` ： 上下文

3.7.2 接口名称：获取全局唤醒开关

```
int getGlobalVwkSwitch()
```

返回值

`1` :开启;

`-1` :关闭

3.7.3 接口名称：设置全局唤醒开关

```
int setGlobalVwkSwitch(int value)
```

接口参数

value : 1：开启；-1：关闭

返回值

0：成功；非0：失败

3.7.4 接口名称：设置全局唤醒开关监听

```
void setGlobalVwkObserver(VoiceSettingObserver observer)
```

接口参数

observer :监听器

3.7.5 接口名称：获取自定义唤醒开关

```
int getCustomVwkSwitch()
```

返回值

1 :开启;

-1 :关闭

3.7.6 接口名称：设置自定义唤醒开关

```
int setCustomVwkSwitch(int value)
```

接口参数

value : 1：开启；-1：关闭

返回值

0：成功；非0：失败

3.7.7 接口名称：设置自定义唤醒开关监听

```
void setCustomVwkSwitchObserver(VoiceSettingObserver observer)
```

接口参数

observer :监听器

3.7.8 接口名称：获取oneshot开关

```
int getOneshotSwitch()
```

返回值

1 :开启;

-1 :关闭

3.7.9 接口名称：设置oneshot开关

```
int setOneshotSwitch(int value)
```

接口参数

value : 1：开启；-1：关闭

返回值

0：成功；非0：失败

3.7.10 接口名称：设置oneshot开关监听

```
void setOneshotSwitchObserver(VoiceSettingObserver observer)
```

接口参数

observer :监听器

3.7.11 接口名称：获取场景唤醒开关

```
int getSceneVwkSwitch()
```

返回值

1 :开启;

-1 :关闭

3.7.12 接口名称：设置场景唤醒开关

```
int setSceneVwkSwitch(int value)
```

接口参数

value : 1：开启；-1：关闭

返回值
0：成功；非0：失败

3.7.13 接口名称：设置场景唤醒开关监听

```
void setSceneVwkSwitchObserver(VoiceSettingObserver observer)
```

接口参数

`observer` :监听器

3.7.14 接口名称：获取TTS发音人

```
String getTtsRole()
```

返回值
发音人名称

3.7.15 接口名称：设置TTS发音人

```
int setTtsRole(String role)
```

接口参数

`role` : 发音人名称

返回值
0：成功；非0：失败

3.7.16 接口名称：设置TTS发音人监听

```
void setTtsRoleObserver(VoiceSettingObserver observer)
```

接口参数

`observer` :监听器

3.7.17 接口名称：获取自定义唤醒词

```
String getCustomVwkWord()
```

返回值
自定义唤醒词

3.7.18 接口名称：设置自定义唤醒词

```
int setCustomVwkWord(String vwkWord)
```

接口参数

`vwkWord`：自定义唤醒词

返回值

0：成功；非0：失败

3.7.19 接口名称：设置自定义唤醒词监听

```
void setCustomVwkWordObserver(VoiceSettingObserver observer)
```

接口参数

`observer`:监听器

4 Demo示例

> Demo 工程请参考附件：xxxservicedemo.zip