VCS SWAP SDK 使用说明文档

版本	修改人	修改日期	修改内容
V 1.0	周浩	2022-01-26	1. Markdown 版本改成 Word 版本 2. 删除 NotificationManager 相关使用说明 3. 修改 Service 类参考代码 4. 增加详细指令格式到附录 5. 增加名字解释 6. 规整目录结构
V1.1	周浩	2022-02-10	1. 修改错误的方法 2. 修改对齐格式 3. 增加 2.3.3：VTS 使用说明和代码参考 4. Demo 中增加 TTS 停止代码参考 5. 修改 TtsOption 默认值 6. 增加 2.1.3：msg 格式说明 7. **修改和 VCS 通信返回值的定义** 8. 修改附录中返回值指令格式的描述

SWAP SDK 是 VCS 模块提供给下游 APP 使用系统中语音能力的 aar 包，以 swap-xxx.aar 命名。主要提供以下能力：

1. 接收语音操作指令
2. 使用 TTS 能力
3. 使用可见即可说能力
4. 读取和修改语音设置

1. SDK 使用

1.1 名词解释

VCS	Voice Control Service	语音控制服务
TTS	Text To Speech	文本转语音
VTS	Voice Touch Screen	语音触控屏幕（可见即可说）

1.2 配置工程

获取到的 aar 包放到集成模块的**libs**目录下， 并在模块的**build.gradle**中增加依赖。

注意：swap-xxx.aar 需要和 libs 目录下文件名保存一致。

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

2. SDK 功能说明

swap SDK 需要和 VCS apk 配合使用，swap 和 VCS 通过 aidl 进行通信。

2.1 接收语音操作指令

语音系统下发的指令通过 VCS 进行集中处理，处理后会分发不同的请求到各个应用中，每个应用都需要按照要求实现服务监听 VCS 的请求，把响应回传到 VCS 中。

新建 service 类：

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

    //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 Remo
teException {
            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);
            }
        }
    }

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

    @Override
    public IBinder onBind(Intent intent) {
        return mBind;
    }

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

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

在 AndroidManifest.xml 中注册 XxxService 类

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

注意：VCS 发送的请求和需要的响应都是 json 格式，详细请求和响应参考附录

Service 中收到 VCS 的请求可以通过两种方式返回操作结果给 VCS。

1. 收到请求后同步处理，把处理结果返回给 VCS。

2. 收到请求后异步处理，通过 VCS 注册的监听器返回给 VCS 处理结果。

2.1.1 IBridge.Stub 接口说明

限定符和类型	方法和说明
String	sendRequest(String uuid, String msg) 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.

2.1.2 IBridgeListener 接口说明

限定符和类型	方法和说明
void	onResponse(String uuid, String msg) uuid：返回码，需要和请求码一致 msg：额外信息，可以为 null、json 格式字符串

2.1.3 msg 格式说明

同步返回给 VCS 结果和通过 IBridgeListener 接口返回给 VCS 指令执行结果，msg 的格式一致。

属性|字段	方法和说明
code	执行结果返回码，0：成功；非 0：失败；方便后续扩展
msg	指令执行之后的播报信息。 例如：打开空调指令执行成功， 需要播报提示的内容为：“已为您打开空调”
nluType	执行的指令类型。 空调：3 电量：8

	油量：9 座椅：10 胎压：11
background	标适此次 msg 信息是否需要前台播报。 前台播报：播报内容在页面前台提示用户 后台播报：只播报内容，没有界面提示

2.2 使用 TTS 能力

VCS 内部集成了 TTS 服务，应用可以通过 swap sdk 中提供的 TtsClientManager 类使用 TTS 能力。

2.2.1 TtsClientManager 接口说明

限定符和类型	方法和说明
TtsClientManager	TtsClientManager(@NonNull Context context) context：上下文
TtsClientManager	TtsClientManager(@NonNull Context context, @NonNull String packageName, @NonNull String serviceName) context：上下文 packageName：VCS 包名 serviceName：VCS 中 TTS 服务名
int	startTts(String text, TtsOption option) text：待合成的文本 option：合成文本时的选项
int	startTts(String text, TtsOption option, ITtsListener listener) text：待合成的文本 option：合成文本时的选项 listener：监听器
void	setTtsListener(ITtsListener ttsListener) listener：监听器

2.2.2 TtsOption 说明

属性|字段	方法和说明
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）

2.2.3 ITtsListener 接口说明

限定符和类型	方法和说明
void	onSpeechStart() 开始播放回调
void	onSpeechFinished() 播放结束回调
void	onSpeechInterrupted() 播放被打断
void	onSpeechError(int errorCode) 播放出现错误 errorCode：错误码

2.3 使用 VTS 能力

VCS 内部集成了 VTS 服务，应用可以通过 swap sdk 中提供的 VtsClientManager 类使用 Vts 能力。

2.3.1 VtsClientManager 接口说明

限定符和类型	方法和说明
void	registerBack(java.lang.String name, java.lang.String[] utterance) 控件类型：返回 支持话术：返回 友情提示：指令接收参考：IVtsListener.onBack()
void	registerBuyMembership(java.lang.String name) 控件类型：购买会员 支持话术：买会员、购买会员 友情提示：指令接收参考： IVtsListener.onBuyMember()
void	registerCancel(java.lang.String name) 控件类型：取消 支持话术：取消、点击取消 友情提示：指令接收参考：IVtsListener.onCancel()
void	registerChangeSelect(java.lang.String name, java.lang.String[] utterance) 控件类型：列表控制 支持话术：上一个，下一个，换一个 友情提示：指令接收参考： IVtsListener.onChangeSelect(String)
void	registerCheckOrder(java.lang.String name) 控件类型：查看订单 支持话术：查看 xx 订单，查看订单 友情提示：指令接收参考： IVtsListener.onCheckOrder(String)
void	registerClearCache(java.lang.String name) 控件类型：清除缓存 支持话术：清除缓存、清空缓存 友情提示：指令接收参考： IVtsListener.onClearCache()
void	registerConfirm(java.lang.String name) 控件类型：确认 支持话术：确认、点击确认 友情提示：指令接收参考：IVtsListener.onConfirm()
void	registerDefault() 控件类型：默认话术[第 X 个]当 X 超出范围时的兜底控件 支持话术：第 X 个，X 为超出范围的数值） 友情提示：指令接收参考：IVtsListener.onDefault()
void	registerDelete(java.lang.String name, java.lang.String[] utterances, int index) 控件类型：删除 支持话术：删除 xxx，删除第 x 项 友情提示：指令接收参考： IVtsListener.onDelete(int, String)
void	registerExpand(java.lang.String name) 控件类型：展开 支持话术：查看更多，查看更多结果 友情提示：指令接收参考： IVtsListener.onExpand()
void	registerFavorite(java.lang.String name, int index, java.lang.String[] utterances) 控件类型：收藏 支持话术：收藏(取消收藏)xxx，收藏（取消收藏）第几项 友情提示：指令接收参考： IVtsListener.onFavorite(String, int, String)
void	registerFilter() 控件类型：筛选 支持话术：筛选/只看 + 【筛选条件】 友情提示：指令接收参考： IVtsListener.onFilter(String)

void	registerInput(java.lang.String type) 控件类型：输入 支持话术：时间输入 xxx，地址输入 xxx 友情提示：指令接收参考：IVtsListener.onInput(String, String, String)
void	registerLogin(java.lang.String name) 控件类型：登录 支持话术：去登录、登录、登录 xxx 友情提示：指令接收参考：IVtsListener.onLogin()
void	registerNavigation(java.lang.String name, java.lang.String[] utterances, int index) 控件类型：导航 支持话术：导航过去/导航去这里/开始导航，导航到 xxx，导航第 x 个 友情提示：指令接收参考：IVtsListener.onNavigation(String, int)
void	registerPager() 控件类型：翻页 支持话术：打开（上/下）一页，打开第 X 页，打开（首页/最后一页） 友情提示：指令接收参考：IVtsListener.onPager(String, int)
void	registerPay(java.lang.String name) 控件类型：确认支付 支持话术：去支付、确认支付 友情提示：指令接收参考：IVtsListener.onPay()
void	registerPhone(java.lang.String name, java.lang.String[] utterances, int index) 控件类型：电话 支持话术：打电话给 xxx 友情提示：指令接收参考：IVtsListener.onPhoneCall(int, String)
void	registerRanker(java.lang.String type) 控件类型：排序 支持话术：按[销量/价格/面积/距离..]排序 按[销量/价格/面积/距离..]
void	registerSearch(java.lang.String name) 控件类型：搜索 支持话术：搜 xxx 友情提示：指令接收参考：IVtsListener.onSearch(String)
void	registerSelect(java.lang.String name, int index) 控件类型：点击，选择 支持话术：点击[name]，选择[name]，选择第[index]个 友情提示：指令接收参考：IVtsListener.onSelect(String, int)
void	registerSelect(java.lang.String name, int index, java.lang.String[] utterances) 控件类型：点击，选择 支持话术：点击[name]，选择[name]，选择第[index]个 友情提示：指令接收参考：IVtsListener.onSelect(String, int)
void	registerTab(java.lang.String tabName, int index) 控件类型：Tab 支持话术：切换到[TabName]/切换为[TabName] 友情提示：指令接收参考：IVtsListener.onTabSelect(String, int)
void	registerTab(java.lang.String tabName, int index, java.lang.String[] utterances) 控件类型：Tab 支持话术：切换到[TabName]/切换为[TabName] 友情提示：指令接收参考：
void	registerTab(java.lang.String tabName, int index, java.lang.String[] utterances) 控件类型：Tab 支持话术：切换到[TabName]/切换为[TabName] 友情提示：指令接收参考：
void	registerTab(java.lang.String tabName, int index, java.lang.String[] utterances) 控件类型：Tab 支持话术：切换到[TabName]/切换为[TabName] 友情提示：指令接收参考：
void	enableVtsCapability() 可见即可说功能生效(使能可见即可说)

void	disableVtsCapability() 可见即可说功能失效(使不能可见即可说)
void	addVtsListener(IVtsListener listener) 增加可见即可说监听 接口参数 listener: 监听器
void	deleteVtsListener(IVtsListener listener) 删除可见即可说监听 接口参数 listener: 监听器

2.3.2 IVtsListener 接口说明

限定符和类型	方法和说明
void	onBack() 收到指令：返回；当使用 VtsClientManager .registerBack(String, String[])注册的能力触发时调用此方法
void	onBuyMember() 收到指令：购买会员；当使用 VtsClientManager .registerBuyMembership(String)注册的能力触发时调用此方法
void	onCancel() 收到指令：取消、点击取消；当使用(String) 注册的能力触发时调用此方法
void	onChangeSelect(java.lang.String select) 收到指令：下一个、上一个、换一个；当使用 VtsClientManager .registerChangeSelect(String, String[]) 注册的能力触发时调用此方法
void	onCheckOrder(java.lang.String type) 收到指令：查看 xx 订单，查看订单；当使用 VtsClientManager .registerCheckOrder(String) 注册的能力触发时调用此方法
void	onClearCache() 收到指令：清除缓存；当使用 VtsClientManager .registerClearCache(String) 注册的能力触发时调用此方法
void	onConfirm() 收到指令：确认、点击确认；当使用 VtsClientManager .registerConfirm(String) 注册的能力触发时调用此方法
void	onDefault() 收到指令：[第 X 个]当 X 超出范围时的回调指令；当使用 VtsClientManager .registerDefault() 注册的能力触发时调用此方法

void	onDelete(int index, java.lang.String name) 收到指令：删除 xxx，删除第 x 项；当使用 VtsClientManager .registerDelete(String, String[], int) 注册的能力触发时调用此方法
void	onExpand() 收到指令：查看更多，查看更多结果； 当使用 VtsClientManager .registerExpand(String) 注册的能力触发时调用此方法
void	onFavorite(java.lang.String type, int index, java.lang.String name) 收到指令：收藏(取消收藏) xxx，收藏（取消收藏）第几项；当使用 VtsClientManager .registerFavorite(String, int, String[]) 注册的能力触发时调用此方法
void	onFilter(java.lang.String content) 收到指令：筛选/只看 +[筛选条件]；当使用 VtsClientManager .registerFilter()注册的能力触发时调用此方法
void	onInput(java.lang.String content, java.lang.String type, java.lang.String time) 收到指令：时间输入 xxx，地址输入 xxx； 当使用 VtsClientManager .registerInput(String) 注册的能力触发时调用此方法
void	onLogin() 收到指令：去登录、登录、登录 xxx；当使用 VtsClientManager .registerLogin(String)注册的能力触发时调用此方法
void	onNavigation(java.lang.String content, int index) 收到指令：导航过去/导航去这里/开始导航，导航到 xxx，导航第 x 个；当使用 VtsClientManager .registerNavigation(String, String[], int)注册的能力触发时调用此方法
void	onPager(java.lang.String event, int value) 当使用 VtsClientManager .registerPager()注册的能力触发时调用此方法
void	onPay() 收到指令：去支付、确认支付；当使用 VtsClientManager .registerPay(String) 注册的能力触发时调用此方法
void	onPhoneCall(int index, java.lang.String name) 收到指令：打电话给 xxx；当使用 VtsClientManager .registerPhone(String, String[], int)注册的能力触发时调用此方法
void	onRanker(java.lang.String type, java.lang.String range) 收到指令：按[销量/价格/面积/距离..]排序 按[销量/价格/面积/距离. .]
void	onScroll(java.lang.String direction, java.lang.String event, int value) 当使用 VtsClientManager .registerScroll()注册的能力触发时调用此方法
void	onSearch(java.lang.String content) 收到指令：搜索 xxx；当使用 VtsClientManager .registerSearch(String) ()}注册的能力触发时调用此方法

void	onSelect(java.lang.String name, int index) 当使用 VtsClientManager.registerSelect(String, int) 注册的能力触发时调用此方法；content 不存在时为 null，index 不存在时为<=0 的值
void	onTabSelect(java.lang.String name, int index) 当使用 VtsClientManager.registerTab(java.lang.String, int)注册的能力触发时调用此方法；回调此方法；

2.3.3 Vts 使用说明

应用在前台，需要 VTS 能力时候相继调用注册能力、设置监听(非每次必须)、使能接口，从监听器中接收用户喊的词。

```
1.  public void testVTS(View view) {
2.      Log.d(TAG, "testVTS: ");
3.      //注册 VTS 能力
4.      mVtsClientManager.registerSelect("确定", 1);
5.      mVtsClientManager.registerSelect("取消", 2);
6.      mVtsClientManager.registerSelect("返回", 3);
7.      //增加回调监听
8.      mVtsClientManager.addVtsListener(vtsListener);
9.      //VTS 使能,使能之后，唤醒语音助手，喊注册的词，从监听器中接收回调
10.     mVtsClientManager.enableVtsCapability();
11. }
```

应用不可见（不需要可见即可说能力）的时候应当删除注册的 VTS 监听，防止当前在前台的应用注册的 VTS 能力触发后发送给自己，再关闭 VTS 能力。

```
1.  @Override
2.  protected void onPause() {
3.      super.onPause();
4.      //删除 VTS 监听器，防止别人注册的 VTS 能力触发后发送给自己消息
5.      mVtsClientManager.deleteVtsListener(vtsListener);
6.      //关闭 VTS 能力
7.      mVtsClientManager.disableVtsCapability();
8.  }
```

2.4 读取和修改语音设置

swap sdk 内部封装了语音设置的读取和修改接口，外部应用可以通过 sdk 提供的 VoiceSettingManager 接口读取和修改语音设置项。

限定符和类型	方法和说明

int	getGlobalVwkSwitch() 获取全局唤醒开关 返回值 1:开启； -1:关闭
int	setGlobalVwkSwitch(int value) 设置全局唤醒开关 接口参数 value：1：开启；-1：关闭 返回值 0：成功；非 0：失败
void	setGlobalVwkObserver(VoiceSettingObserver observer) 设置全局唤醒开关监听 接口参数 observer:监听器
int	getCustomVwkSwitch() 获取自定义唤醒开关 返回值 1:开启； -1:关闭
int	setCustomVwkSwitch(int value) 设置自定义唤醒开关 接口参数 value：1：开启；-1：关闭 返回值 0：成功；非 0：失败
void	setCustomVwkSwitchObserver(VoiceSettingObserver observer) 设置自定义唤醒开关监听 接口参数 observer:监听器
int	getOneshotSwitch() 获取 oneshot 开关 返回值 1:开启； -1:关闭
int	setOneshotSwitch(int value) 设置 oneshot 开关 接口参数 value：1：开启；-1：关闭 返回值

	0：成功；非 0：失败
void	setOneshotSwitchObserver(VoiceSettingObserver observer) 设置 oneshot 开关监听 接口参数 observer:监听器
int	getSceneVwkSwitch() 获取场景唤醒开关 返回值 1:开启； -1:关闭
int	setSceneVwkSwitch(int value) 设置场景唤醒开关 接口参数 value: 1：开启；-1：关闭 返回值 0：成功；非 0：失败
void	setSceneVwkSwitchObserver(VoiceSettingObserver observer) 接口参数 observer:监听器
String	getTtsRole() 获取 TTS 发音人 返回值 发音人名称
int	setTtsRole(String role) 接口参数 role: 发音人名称 返回值 0：成功；非 0：失败
void	setTtsRoleObserver(VoiceSettingObserver observer) 设置 TTS 发音人监听 接口参数 observer:监听器
String	getCustomVwkWord() 获取自定义唤醒词 返回值 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
void	setCustomVwkWordObserver(VoiceSettingObserver observer) 接口参数 observer:监听器

3. 附录

3.1 车控车设模块

3.1.1. 打开/关闭 空调/AC

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setHvacState",
    "param":{
        "position":1,
        "state":true
    }}
```

position：位置信息

- 0x03：前排
- 0x0C：后排

state:状态

- true：打开
- false：关闭

结果格式：

```
{
    "code":0,
    "msg":"需要播报的文本",
```

```
    "nluType":3,
    "background":false
    }
```

3.1.2. 打开/关闭 MAX AC

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setMaxAcState",
    "param":{
        "state":true
    }}
```

state:状态

- true： 打开
- false： 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.3. 打开/关闭 Auto 模式

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setAutoMode",
    "param":{
        "position":1,
        "state":true
    }}
```

position：位置信息

- 0x03： 前排
- 0x0C： 后排

state:状态

- true： 打开

- false： 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.4. 打开/关闭 同步模式

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setSyncState",
    "param":{
        "state":true
    }}
```

state:状态

- true： 打开
- false： 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.5. 打开/关闭 后排空调锁

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setRearPanelLock",
    "param":{
        "state":true
    }}
```

state:状态

- true： 打开
- false： 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.6. 打开/关闭 最大前除霜

```
{
```

```
    "module":"VEHICLE_CONTROL",

    "function":"setMaxDefrostState",

    "param":{

        "state":true

    }}
```

state:状态

- true： 打开
- false： 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.7. 设置空调循环模式

```
{

    "module":"VEHICLE_CONTROL",

    "function":"setCycleMode",

    "param":{

        "mode":1

    }}
```

mode:模式

- 1： 内循环
- 2： 外循环

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.8. 调节空调温度

```
{

    "module":"VEHICLE_CONTROL",

    "function":"adjustTemperature",

    "param":{

        "position":1,

        "shift":-3
```

```
    }}
```

position：位置信息

- 0x01：主驾
- 0x02：副驾
- 0x03：前排
- 0x0C：后排
- 0x0F：全车

shift:偏移值

- 正值 ：调高温度
- 负值 ：调低温度

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.9. 设置空调温度

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setTemperature",
    "param":{
        "position":1,
        "target":24
    }}
```

position：位置信息

- 0x01：主驾
- 0x02：副驾
- 0x03：前排
- 0x0C：后排
- 0x0F：全车

target: 温度值

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.10. 调整空调风速

```
{
    "module":"VEHICLE_CONTROL",
```

```
    "function":"adjustFanSpeed",
    "param":{
        "position":1,
        "shift":-2
    }}
```

position：位置信息

- 0x01：主驾
- 0x02：副驾
- 0x03：前排
- 0x0C：后排
- 0x0F：全车

shift:调整偏移值

- 正值 ：调高风速
- 负值 ：调低风速

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.11. 设置空调风速

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setFanSpeed",
    "param":{
        "position":1,
        "target":3
    }}
```

position：位置信息

- 0x01：主驾
- 0x02：副驾
- 0x03：前排
- 0x0C：后排
- 0x0F：全车

target:空调目标风速

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.12. 设置空调吹风模式

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setBlowerMode",
    "param":{
        "position":1,
        "mode":1
    }}
```

position：当前此参数无用

mode:吹风模式

- 0x01：吹窗
- 0x02：吹面
- 0x04：吹脚
- 0x03：吹面 + 吹窗
- 0x05：吹窗 + 吹脚
- 0x06：吹面 + 吹脚

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.13. 打开/关闭 智能馨风

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setIntellectWind",
    "param":{
        "state":true
    }}
```

state:状态

- true：打开
- false：关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.14. 打开/关闭 座舱新风

```
{

    "module":"VEHICLE_CONTROL",

    "function":"setCabinFreshAir",

    "param":{

        "state":true

    }}
```

state:状态

- true： 打开
- false： 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.15. 打开/关闭 电动出风口

```
{

    "module":"VEHICLE_CONTROL",

    "function":"setOutletEnable",

    "param":{

        "position":1,

        "state":true

    }}
```

position：位置信息

- 0x01： 主驾
- 0x02： 后排
- 0x03： 主副驾

state:状态

- true： 打开
- false： 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.16. 设置电动出风口吹风模式

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setOutletMode",
    "param":{
        "position":1,
        "mode":1
    }}
```

position：位置信息

- 0x01：主驾
- 0x02：副驾
- 0x03：主副驾

mode:吹风模式

- 0x1：上下扫风
- 0x2：左右扫风
- 0x3：全域扫风
- 0x4：朝人吹风
- 0x3：避人吹风

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.17. 打开/关闭 方向盘加热

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setSteeringWheelHeatState",
    "param":{
        "state":1
    }}
```

state:开关（方向盘加热等级，预留）

- 0：关闭
- 1：打开

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.18. 打开/关闭 座椅加热

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setSeatHeatState",
    "param":{
        "position":1,
        "state":1
    }}
```

position：位置信息

- 0x01：主驾
- 0x02：副驾
- 0x03：主副驾

state:开关或加热等级

- 0：关闭
- 1：打开或 1 档

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.19. 打开/关闭 座椅通风

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setSeatVentilateState",
    "param":{
        "position":1,
        "state":1
    }}
```

position：位置信息

- 0x01：主驾

- 0x02：副驾
- 0x03：主副驾

state:通风状态，通风等级

- 0：关闭
- 1：打开或 1 级别

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.20. 打开/关闭 座椅按摩

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setSeatMassageState",
    "param":{
        "position":1,
        "state":1
    }}
```

position：位置信息

- 0x01：主驾
- 0x02：副驾
- 0x03：主副驾

state:座椅按摩状态

- 0：关闭
- 1：打开

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.21. 调整座椅

```
{
    "module":"VEHICLE_CONTROL",
    "function":"adjustSeatState",
    "param":{
        "position":12,
        "mode":1,
```

```
        "value":0

    }}
```

position：位置信息

- 0x0C：二排（目前需求中只有二排座椅调节）

mode：调整部位

- 0x01：座椅上下调
- 0x02：座椅前后调
- 0x04 ：座椅靠背前后调

value：调整值

- 负值：向下或向后
- 0：复位
- 正值：向上或向前

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.22. 打开/关闭 氛围灯

```
{

    "module":"VEHICLE_CONTROL",

    "function":"setAtmosphereLampState",

    "param":{

        "state":true

    }}
```

state:状态

- true：打开
- false：关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.23. 调整氛围灯亮度

```
{

    "module":"VEHICLE_CONTROL",

    "function":"adjustAtmosphereLampBrightness",
```

```
    "param":{

        "shift":-20

    }}
```

shift:调整值

- 正值 ：亮度调高
- 负值 ：亮度调低

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.24. 设置氛围灯亮度

```
{

    "module":"VEHICLE_CONTROL",

    "function":"setAtmosphereLampBrightness",

    "param":{

        "target":100

    }}
```

target:目标亮度值

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.25. 设置氛围灯颜色

```
{

    "module":"VEHICLE_CONTROL",

    "function":"setAtmosphereLampColor",

    "param":{

        "color":6,

    }}
```

color:颜色值

- 0x00：顺序切换颜色
- 0x01：冰蓝色
- 0x02：橙色

- 0x03：浅蓝色
- 0x04：红色
- 0x05：绿色
- 0x06：深蓝色
- 0x07：紫色

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.26. 打开/关闭 香氛系统

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setFragranceState",
    "param":{
        "state":true
    }}
```

state:状态

- true：打开
- false：关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.27. 调节香氛浓度

```
{
    "module":"VEHICLE_CONTROL",
    "function":"adjustFragranceConcentration",
    "param":{
        "shift":1
    }}
```

shift:调节值

- 正值 ：调高浓度
- 负值 ：调低浓度

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.28. 设置香氛浓度

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setFragranceConcentration",
    "param":{
        "target":3
    }}
```

target:浓度等级（1~3）

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.29. 设置香氛气味类型

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setFragranceType",
    "param":{
        "value":1
    }}
```

value:香味类型（0~3）

- 0：顺序切换
- 1：香味一
- 2：香味二
- 3：香味三

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.30. 天窗操作

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setSunRoofState",
```

```
    "param":{

        "state":100,

    }}
```

state: 天窗状态

- 0：关闭
- 50：半开
- 100：全开
- 负值 ：翘起(预留)

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.31. 天窗遮阳板操作

```
{

    "module":"VEHICLE_CONTROL",

    "function":"setSunShadeState",

    "param":{

        "state":0,

    }}
```

state: 天窗遮阳板状态

- 0：关闭
- 50：半开
- 100：全开

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.32. 打开/关闭 后备箱

```
{

    "module":"VEHICLE_CONTROL",

    "function":"setCarBootState",

    "param":{

        "state":0

    }}
```

state:状态值

- 0：关闭
- 100：打开

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.33. 打开/关闭 前舱盖

```
{
    "module":"VEHICLE_CONTROL",
    "function":"setForeHatchState",
    "param":{
        "state":0
    }}
```

state:前舱盖状态值

- 0：关闭
- 100：打开

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.34. 查询汽车电量

```
{
    "module":"VEHICLE_CONTROL",
    "function":"queryVehicleBatterMessage",
    "param":{
    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.35. 查询汽车油量

```
{
    "module":"VEHICLE_CONTROL",
    "function":"queryVehicleFuelMessage",
```

```
    "param":{

    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.36. 查询胎压信息

```
{

    "module":"VEHICLE_CONTROL",

    "function":"queryVehicleTirePressure",

    "param":{

        "position":12

    }}
```

position：位置信息

- 0x01：左前轮
- 0x02：右前轮
- 0x03：前轮
- 0x04：左后轮
- 0x08：右后轮
- 0x05：左侧车轮
- 0x0A：右侧车轮
- 0x0F：全车车辆

结果格式：参考 2.1.3 和 3.1.1 章节

3.1.37. 查询违章信息

```
{

    "module":"VEHICLE_CONTROL",

    "function":"queryVehicleViolationRecord",

    "param":{

    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.2 系统设置模块

3.2.1. 打开/关闭 蓝牙

```
{
    "module":"SYSTEM_SETTING",
    "function":"setBluetoothState",
    "param":{
        "position":1,
        "state":true
    }}
```

position：位置信息

- 0x01：主驾
- 0x02：副驾
- 0x03：主副驾

state:

- true：打开
- false：关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.2. 打开/关闭 WIFI

```
{
    "module":"SYSTEM_SETTING",
    "function":"setWifiState",
    "param":{
        "state":false
    }}
```

state:

- true：打开
- false：关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.3. 打开/关闭 车辆热点

```
{

    "module":"SYSTEM_SETTING",

    "function":"setHotspotState",

    "param":{

        "state":true

    }}
```

state:

- true：打开
- false：关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.4. 更换主题

```
{

    "module":"SYSTEM_SETTING",

    "function":"changeTheme",

    "param":{

        "theme":0

    }}
```

theme: 主题。0：切换主题；1、2、3、4... 主题代号。

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.5. 打开/关闭 屏幕

```
{

    "module":"SYSTEM_SETTING",

    "function":"openScreen",

    "param":{
```

```
        "state":true

    }}
```

state:

- true : 打开
- false : 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.6. 调整屏幕亮度

```
{

    "module":"SYSTEM_SETTING",

    "function":"adjustScreenBrightness",

    "param":{

        "shift":-8

    }}
```

value：亮度调整值；正值：调高亮度；负值：调低亮度

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.7. 设置屏幕亮度

```
{

    "module":"SYSTEM_SETTING",

    "function":"setScreenBrightness",

    "param":{

        "value":60

    }}
```

value：亮度调整值；正值：调高亮度；负值：调低亮度

- 0：最暗
- 100：最亮

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.8. 打开/关闭 分屏

```
{

    "module":"SYSTEM_SETTING",

    "function":"setScreenSplit",

    "param":{

        "state":true

    }}
```

state:

- true： 打开
- false： 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.9. 打开/关闭 精简屏幕

```
{

    "module":"SYSTEM_SETTING",

    "function":"setScreenTidy",

    "param":{

        "position":1,

        "tidy":true

    }}
```

position：位置信息

- 0x01： 主驾
- 0x02： 副驾
- 0x03： 主副驾

state:

- true： 打开
- false： 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.10. 切换主副驾屏幕

```
{
    "module":"SYSTEM_SETTING",
    "function":"exchangeScreen"}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.11. 调整音量

```
{
    "module":"SYSTEM_SETTING",
    "function":"adjustVolume",
    "param":{
        "type":2,
        "value":-20
    }}
```

type: 音量类型

- 1：系统
- 2：多媒体
- 3：导航
- 4：语音

value: 正值：调高音量；负值：调低音量

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.12. 设置音量

```
{
    "module":"SYSTEM_SETTING",
    "function":"setVolume",
    "param":{
        "type":2,
```

```
        "value":-20

    }}
```

type: 音量类型

- 1：系统
- 2：多媒体
- 3：导航
- 4：语音

value: 音量值

- 0：最小
- 100： 最大

结果格式：参考 2.1.3 和 3.1.1 章节

3.2.13. 打开/关闭 系统静音

```
{

    "module":"SYSTEM_SETTING",

    "function":"setSystemMute",

    "param":{

        "state":false

    }}
```

state:

- true : 打开
- false : 关闭

结果格式：参考 2.1.3 和 3.1.1 章节

3.3 蓝牙电话模块

3.3.1. 打开蓝牙电话

```
{

    "module":"BT_PHONE",

    "function":"openBtPhone",
```

```
    "param":{


    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.3.2. 关闭蓝牙电话

```
{
    "module":"BT_PHONE",
    "function":"closeBtPhone",
    "param":{


    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.3.3. 接听蓝牙电话

```
{
    "module":"BT_PHONE",
    "function":"acceptCall",
    "param":{


    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.3.4. 挂断蓝牙电话

```
{
    "module":"BT_PHONE",
    "function":"rejectCall",
    "param":{
```

```
}}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.3.5. 重拨

```
{
    "module":"BT_PHONE",
    "function":"redial",
    "param":{

    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.3.6. 同步联系人

```
{
    "module":"BT_PHONE",
    "function":"syncContacts",
    "param":{

    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.3.7. 查看联系人

```
{
    "module":"BT_PHONE",
    "function":"listContacts",
    "param":{
```

```
}}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.3.8. 查看通话记录

```
{
    "module":"BT_PHONE",
    "function":"listRecentCalls",
    "param":{


    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.3.9. 切换设备

```
{
    "module":"BT_PHONE",
    "function":"changeDevice",
    "param":{


    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.4 电子手册模块

3.4.1. 打开电子说明书

```
{
    "module":"E_MANUAL",
    "function":"openManual",
    "param":{
```

```
}}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.4.2. 关闭电子说明书

```
{
    "module":"E_MANUAL",
    "function":"closeManual",
    "param":{


    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.4.3. 搜索电子说明书

```
{
    "module":"E_MANUAL",
    "function":"searchManual",
    "param":{
        "keyword":"xxxx"
    }}
```

结果格式：参考 2.1.3 和 3.1.1 章节

3.5 Relax 模块

3.5.1. 设置 xxx 模式

```
{
    "module":"RELAX",
    "function":"setRelaxMode",
```

```
    "param":{

        "relaxMode":0

    }

}
```

relaxMode：模式

- 0x00：换一种模式
- 0x01：森林模式/森林绮镜
- 0x02：大海模式/海屿之恋
- 0x03：馥郁暖心
- 0x04：放松模式
- 0x05：天空之境
- 0x06：星辰入海
- 0x07：夏夜萤火
- 0x08：落日黄昏
- 0x09：宇宙星空
- 0x0A：浪漫幻境
- 0x0B：茶韵自然

结果格式：参考 2.1.3 和 3.1.1 章节

3.5.2. 退出 xxx 模式

```
{

    "module":"RELAX",

    "function":"exitRelaxMode",

    "param":{

        "relaxMode":2

    }

}
```

relaxMode：模式

- 0x00：当前模式
- 0x01：森林模式/森林绮镜
- 0x02：大海模式/海屿之恋
- 0x03：馥郁暖心

- 0x04：放松模式
- 0x05：天空之境
- 0x06：星辰入海
- 0x07：夏夜萤火
- 0x08：落日黄昏
- 0x09：宇宙星空
- 0x0A：浪漫幻境
- 0x0B：茶韵自然

结果格式：参考 2.1.3 和 3.1.1 章节

4. Demo

参考提供的 xxxservicedemo.zip 中 demo 工程。