HMI\_SDK接⼝说明文档

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| 编制 | 范强 | 日期 | 2017-6-26 | 保密等级 | 机密 |
| 审核 | 潘亮 | 日期 | 2017-6-26 | PM | 徐小明 |

修改历史

|  |  |  |  |
| --- | --- | --- | --- |
| 序号 | 日期 | 说明 | 责任人 |
| 1 | 2017-6-26 | 出版作成，待评审 | 范强 |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |
|  |  |  |  |

目录

[1 引言 3](#_Toc486262978)

[1.1 背景 3](#_Toc486262979)

[1.2 内容 3](#_Toc486262980)

[1.3 适用范围 3](#_Toc486262981)

[1.4 术语 3](#_Toc486262982)

[1.5 参考资料 3](#_Toc486262983)

[2 框架结构 4](#_Toc486262984)

[2.1 模块结构 4](#_Toc486262985)

[2.2 接口关系 4](#_Toc486262986)

[3 时序图 5](#_Toc486262987)

[3.1 初始化 5](#_Toc486262988)

[3.2 Show画面请求 7](#_Toc486262989)

[4 导出函数 7](#_Toc486262990)

[4.1 作用 7](#_Toc486262991)

[4.2 HMI SDK初始化 7](#_Toc486262992)

[4.3 HMI SDK释放 8](#_Toc486262993)

[5 HMI\_SDK提供的接口 8](#_Toc486262994)

[5.1 AppDataInterface 8](#_Toc486262995)

[5.1.1 接口说明 8](#_Toc486262996)

[5.1.2 成员函数说明 8](#_Toc486262997)

[5.2 AppListInterface 15](#_Toc486262998)

[5.2.1 接口说明 15](#_Toc486262999)

[5.2.2 成员函数说明 15](#_Toc486263000)

[5.3 rpcValueInterface 18](#_Toc486263001)

[5.3.1 接口说明 18](#_Toc486263002)

[5.3.2 成员函数说明 18](#_Toc486263003)

[6 调用侧实现的接口 24](#_Toc486263004)

[6.1 rpcValueInterface 24](#_Toc486263005)

[6.1.1 接口说明 24](#_Toc486263006)

[6.1.2 成员函数说明 25](#_Toc486263007)

[7 数据结构 28](#_Toc486263008)

[7.1 command 28](#_Toc486263009)

[7.1.1 结构说明 28](#_Toc486263010)

[7.1.2 成员字段说明 28](#_Toc486263011)

[7.2 DeviceData 28](#_Toc486263012)

[7.2.1 结构说明 29](#_Toc486263013)

[7.2.2 成员字段说明 29](#_Toc486263014)

[8 枚举类型说明 29](#_Toc486263015)

[8.1 enum ShowType 29](#_Toc486263016)

[8.2 enum TOUCH\_TYPE 30](#_Toc486263017)

[9 宏定义 30](#_Toc486263018)

[9.1 Alert画面执行状态 30](#_Toc486263019)

[9.2 ScrollMessage画面执行状态 30](#_Toc486263020)

[9.3 按钮按下模式 30](#_Toc486263021)

[9.4 ChoiceSet画面执行状态 31](#_Toc486263022)

[9.5 Slider画面执行状态 31](#_Toc486263023)

[9.6 录音画面执行状态 31](#_Toc486263024)

[9.7 TTS执行状态 31](#_Toc486263025)

[10 用法示例 31](#_Toc486263026)

[10.1 RPC数据读取接口用法示例 31](#_Toc486263027)

[10.1.1 取Alert画面数据 31](#_Toc486263028)

1. 引言
   1. 背景

考虑到HMI通信层、Json解析层以及App管理部分变化不大，客户需要定制和修改的主要是UI显示部分，每次开发新的HMI需要重复拷贝除UI部分的代码，容易产生错误并且需要了解该部分涉及到的相关知识和调用流程，为了减少重复部分的开发工作，特提出将前面提及到不变的部分封装为库的形式，并提供接口及接口说明文档供UI开发人员使用和参考。

* 1. 内容

本文档描述的主要内容包括，HMISDK主要的接口关系图、调用流程（时序图）、各类接口的说明以及接口方法的详细描述，包括参数、返回值、注意事项等。

* 1. 适用范围

本文档主要面向开发人员，指导UI界面开发人员如何去调用 HMISDK，协助开发人员更好的使用HMISDK，解说每个接口函数使用的方法及含义，提高开发效率，减少重复工作。

* 1. 术语

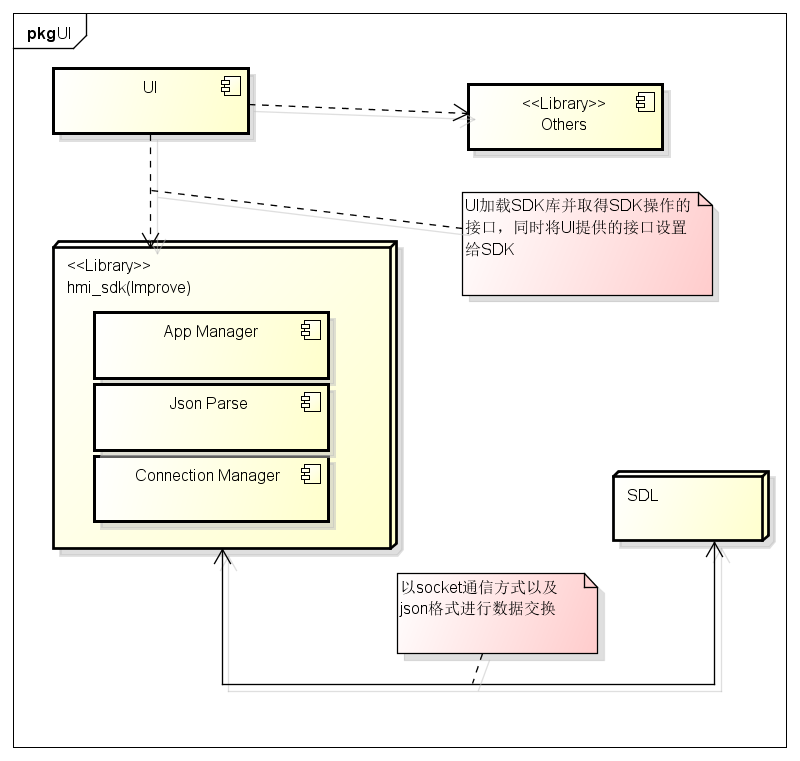
|  |  |
| --- | --- |
| **缩写** | **含义** |
| HMI | 人机交互界面 |
| RPC | 远程过程调用 |
| SDK | HMISDK部分，也是此次重构的重点 |
| UI | 画面显示部分 |
| SDL | SmartDeviceLink |
| VR | 语音识别 |
| TTS | 文本转换语音 |

* 1. 参考资料

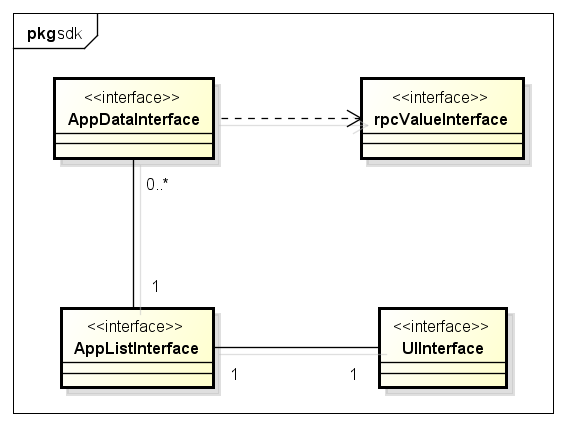
《HMI SDK UI 设计指导说明书.docx》

《HMI\_SDK重构项目需求规格说明书.docx》

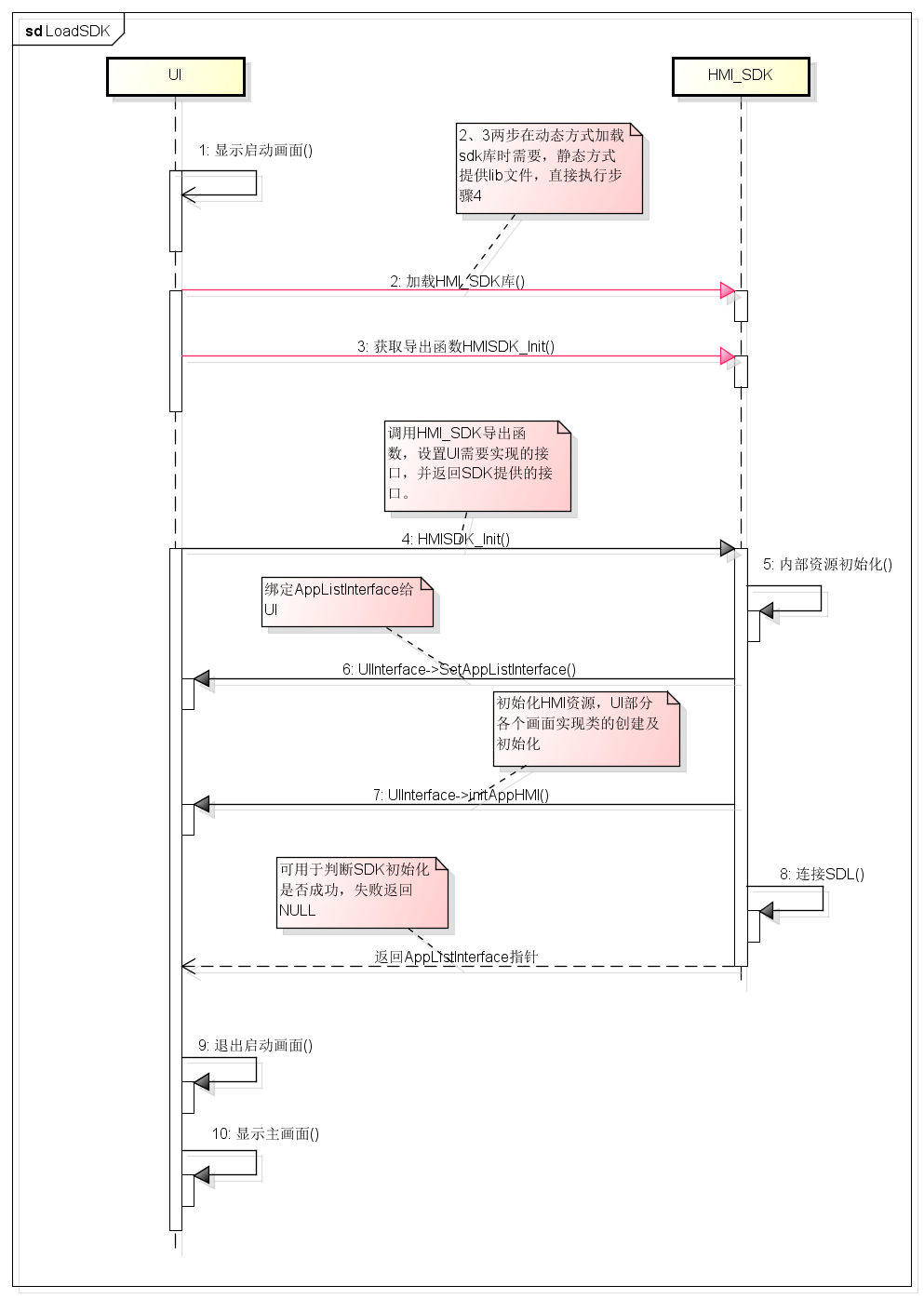
1. 框架结构
   1. 模块结构



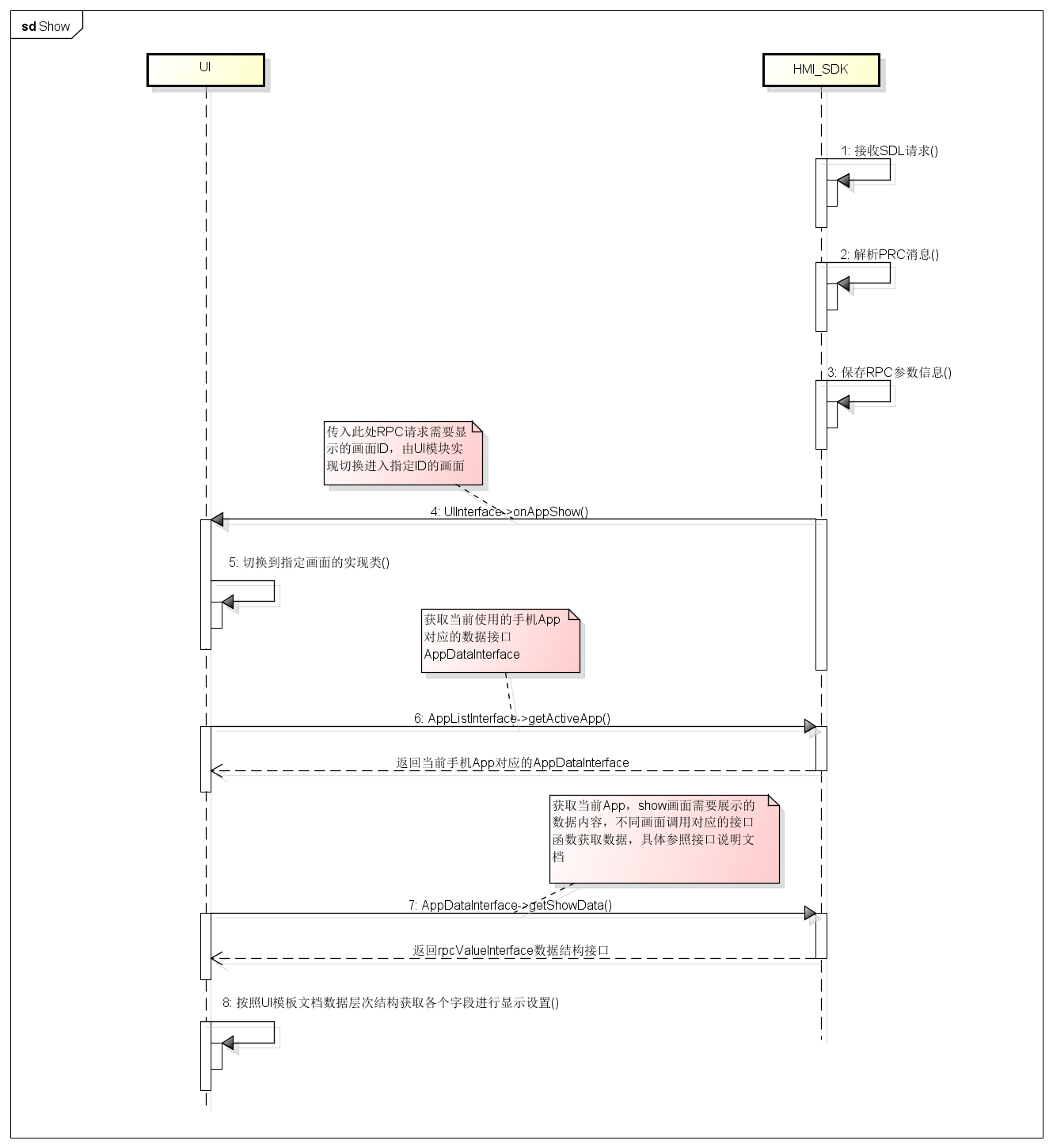
* 1. 接口关系



1. 时序图
   1. 初始化



* 1. Show画面请求



1. 导出函数
   1. 作用

HMI SDK动态库导出函数

* 1. HMI SDK初始化

#### HMISDK\_EXPORT AppListInterface\* HMISDK\_Init (UIInterface \* *pUI*)

HMI SDK初始化

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *pUI* | UIInterface接口 |

##### 返回:

返回值AppListInterface可用于判断SDK初始化是否成功，失败返回NULL

##### 参见:

**UIInterface.h** **AppListInterface.h**

##### 注解:

初始化SDK及HMI资源，并连接SDL

* 1. HMI SDK释放

#### HMISDK\_EXPORT void HMISDK\_UnInit ()

HMI SDK释放

##### 返回:

无

##### 参见:

##### 注解:

释放SDK资源

1. HMI\_SDK提供的接口
   1. AppDataInterface
      1. 接口说明

手机端App数据接口

* + 1. 成员函数说明

#### virtual rpcValueInterface& AppDataInterface::getAlertJson ()[pure virtual]

获取Alert画面需要的rpc数据

##### 返回:

**rpcValueInterface**&

##### 参见:

##### 注解:

#### virtual std::string AppDataInterface::getAppIconFile ()[pure virtual]

获取当前手机端App图标存放路径

##### 返回:

std::string 当前手机端App图标存放路径

##### 参见:

##### 注解:

#### virtual std::string AppDataInterface::getAppName ()[pure virtual]

获取当前手机端App名称

##### 返回:

std::string 当前手机端App名称

##### 参见:

##### 注解:

#### virtual rpcValueInterface& AppDataInterface::getAudioPassThruJson ()[pure virtual]

获取AudioPassThru画面需要的rpc数据

##### 返回:

**rpcValueInterface**&

##### 参见:

##### 注解:

#### virtual std::vector<SMenuCommand> AppDataInterface::getCommandList ()[pure virtual]

获取所有菜单命令项数据

##### 返回:

std::vector<SMenuCommand>

##### 参见:

##### 注解:

#### virtual std::vector<SMenuCommand> AppDataInterface::getCommandList (int *subMenuID*)[pure virtual]

获取指定菜单下所有菜单命令项数据

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *subMenuID* | 菜单标识 |

##### 返回:

std::vector<SMenuCommand>

##### 参见:

##### 注解:

#### virtual int AppDataInterface::getCurUI ()[pure virtual]

获取当前手机端App显示画面的ID

##### 返回:

int 当前手机端App显示画面的ID

##### 参见:

**AppCommon.h** ShowType枚举值定义

##### 注解:

#### virtual rpcValueInterface& AppDataInterface::getInteractionJson ()[pure virtual]

获取ChoiceSet画面需要的rpc数据

##### 返回:

**rpcValueInterface**&

##### 参见:

##### 注解:

#### virtual rpcValueInterface& AppDataInterface::getMediaClockJson ()[pure virtual]

获取MediaShow画面需要的rpc数据

##### 返回:

**rpcValueInterface**&

##### 参见:

##### 注解:

Show画面在media模式时，可通过SetMediaClockTimer rpc请求设置媒体播放进度栏

#### virtual rpcValueInterface& AppDataInterface::getScrollableMsgJson ()[pure virtual]

获取ScrollableMessage画面需要的rpc数据

##### 返回:

**rpcValueInterface**&

##### 参见:

##### 注解:

#### virtual rpcValueInterface& AppDataInterface::getShowData ()[pure virtual]

获取Show画面需要的rpc数据

##### 返回:

**rpcValueInterface**&

##### 参见:

##### 注解:

#### virtual rpcValueInterface& AppDataInterface::getSlider ()[pure virtual]

获取Slider画面需要的rpc数据

##### 返回:

**rpcValueInterface**&

##### 参见:

##### 注解:

#### virtual std::string AppDataInterface::getUrlString ()[pure virtual]

获取VideoStream数据源地址

##### 返回:

std::string 视频流url源地址

##### 参见:

##### 注解:

保留

#### virtual void AppDataInterface::OnAlertResponse (int *reason*)[pure virtual]

Alert画面执行的结果通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *reason* | 执行结果，参照RESULT\_CODE |

##### 返回:

无

##### 参见:

**AppCommon.h** **RESULT\_CODE**

##### 注解:

将Alert画面执行的结果通知给SDK，然后回到前一画面

#### virtual void AppDataInterface::OnCommandClick (int *cmdID*)[pure virtual]

菜单命令项点击通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *cmdID* | 菜单命令项ID |

##### 返回:

无

##### 参见:

##### 注解:

用户在菜单命令画面点击菜单命令项时，调用SDK该函数将通知传递给手机端App

#### virtual void AppDataInterface::OnPerformAudioPassThru (int *code*)[pure virtual]

录音执行的结果通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *code* | 执行结果 |

##### 返回:

无

##### 参见:

**ProtocolDefines.h**

##### 注解:

将录音执行的结果通知给SDK，该函数保留

#### virtual void AppDataInterface::OnPerformInteraction (int *code*, int *choiceID*, bool *bVR* = false)[pure virtual]

ChoiceSet执行的结果通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *code* | 执行结果 |

##### 返回:

无

##### 参见:

**ProtocolDefines.h**

##### 注解:

将ChoiceSet执行的结果通知给SDK，该函数保留

#### virtual void AppDataInterface::OnScrollMessageResponse (int *reason*)[pure virtual]

ScrollMessage画面执行的结果通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *reason* | 执行结果，参照RESULT\_CODE |

##### 返回:

无

##### 参见:

**AppCommon.h** **RESULT\_CODE**

##### 注解:

将ScrollMessage画面执行的结果通知给SDK，然后回到前一画面

#### virtual void AppDataInterface::OnSetMediaClockTimerResponse (int *iCode*)[pure virtual]

SetMediaClockTimer执行的结果通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *code* | 执行结果，参照RESULT\_CODE |

##### 返回:

无

##### 参见:

**AppCommon.h** **RESULT\_CODE**

##### 注解:

将SetMediaClockTimer执行的结果通知给SDK

#### virtual void AppDataInterface::OnShowCommand ()[pure virtual]

请求展示当前App菜单列表画面

##### 返回:

无

##### 参见:

**AppCommon.h** ShowType枚举值定义

##### 注解:

会调用显示ID\_COMMAND画面

#### virtual void AppDataInterface::OnSliderResponse (int *code*, int *sliderPosition*)[pure virtual]

Slider画面执行的结果通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *reason* | 执行结果，参照RESULT\_CODE |

##### 返回:

无

##### 参见:

**AppCommon.h** **RESULT\_CODE**

##### 注解:

将Slider画面执行的结果通知给SDK，然后回到前一画面

#### virtual void AppDataInterface::OnSoftButtonClick (int *sbID*, int *mode*, std::string *strName* = "")[pure virtual]

软按钮点击通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *sbID* | 软按钮ID |
| in | *mode* | 点击模式 |
| in | *strName* | 软按钮名称 |

##### 返回:

无

##### 参见:

**ProtocolDefines.h** **BUTTON\_SHORT** **BUTTON\_LONG**

##### 注解:

用户点击画面软按钮时，调用SDK该函数将通知传递给手机端App

#### virtual void AppDataInterface::OnTTSSpeek (int *code*)[pure virtual]

TTSSpeek执行的结果通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *code* | 执行结果 |

##### 返回:

无

##### 参见:

**ProtocolDefines.h**

##### 注解:

将TTSSpeek执行的结果通知给SDK，该函数保留

#### virtual void AppDataInterface::OnVideoScreenTouch (TOUCH\_TYPE *touch*, int *x*, int *y*)[pure virtual]

VideoStream画面点击移动操作通知

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *touch* | 软按钮ID |
| in | *x* | 鼠标操作的屏幕坐标x轴位置 |
| in | *y* | 鼠标操作的屏幕坐标y轴位置 |

##### 返回:

无

##### 参见:

**ProtocolDefines.h** **TOUCH\_TYPE**

##### 注解:

VideoStream画面点击移动操作时通知SDK，调用SDK该函数将通知传递给手机端App

* 1. AppListInterface
     1. 接口说明

手机端App数据控制接口

* + 1. 成员函数说明

#### virtual void AppListInterface::appUnregistered (int *appId*)[pure virtual]

通知取消App注册

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *appId* | app对应的id |

##### 返回:

无

##### 参见:

##### 注解:

App取消注册时会触发该事件，SDK会将该App数据销毁，并将画面切换到App列表

#### virtual AppDataInterface\* AppListInterface::getActiveApp ()[pure virtual]

获取当前AppDataInterface数据接口

##### 返回:

AppDataInterface\* 当前AppDataInterface数据接口

##### 参见:

##### 注解:

在展示某个画面时，需要获取当前活动App，然后取得画面的数据

#### virtual void AppListInterface::getAppList (std::vector< int > & *vAppIDs*, std::vector< std::string > & *vAppNames*)[pure virtual]

获取所有App信息

##### 参数:

|  |  |  |
| --- | --- | --- |
| out | *vAppIDs* | App标识列表 |
| out | *vAppNames* | App显示名称列表 |

##### 返回:

无

##### 参见:

##### 注解:

获取所有App标识和名称

#### virtual void AppListInterface::getAppList (std::vector< int > & *vAppIDs*, std::vector< std::string > & *vAppNames*, std::vector< std::string > & *vIconPath*)[pure virtual]

获取所有App信息

##### 参数:

|  |  |  |
| --- | --- | --- |
| out | *vAppIDs* | App标识列表 |
| out | *vAppNames* | App显示名称列表 |
| out | *vIconPath* | App显示图标列表 |

##### 返回:

无

##### 参见:

##### 注解:

获取所有App标识、名称及图标

#### virtual void AppListInterface::getDeviceList (std::vector< DeviceData > & *vDevice*)[pure virtual]

获取当前连接所有设备信息

##### 参数:

|  |  |  |
| --- | --- | --- |
| out | *vDevice* | 当前连接所有设备信息 |

##### 返回:

无

##### 参见:

DeviceData结构定义

##### 注解:

#### virtual void AppListInterface::OnAppActivated (int *appID*)[pure virtual]

通知用户画面中选中某个App

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *appID* | App标识 |

##### 返回:

无

##### 参见:

##### 注解:

通知SDK用户在App列表中点击某个App

#### virtual void AppListInterface::OnAppExit ()[pure virtual]

通知退出当前App

##### 返回:

无

##### 参见:

##### 注解:

通知SDK用户在画面点击退出App菜单等操作

#### virtual void AppListInterface::OnDeviceSelect (const std::string *id*)[pure virtual]

通知选择了设备并要求获取设备支持的App信息

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *id* | 设备ID |

##### 返回:

无

##### 参见:

##### 注解:

用户点击画面选择某个设备触发该事件，通知选择了设备并要求获取设备支持的App信息

#### virtual void AppListInterface::OnShowDeviceList ()[pure virtual]

通知需要获取设备列表

##### 返回:

无

##### 参见:

##### 注解:

用户点击画面获取设备列表触发该事件，通知SDK获取设备列表

#### virtual void AppListInterface::OnStartDeviceDiscovery ()[pure virtual]

请求SDK扫描设备信息

##### 返回:

无

##### 参见:

##### 注解:

通知SDK扫描获取有效的设备

#### virtual void AppListInterface::setUIManager (UIInterface \* *pUI*)[pure virtual]

设置UIInterface接口

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *pUI* | UIInterface接口 |

##### 返回:

无

##### 参见:

##### 注解:

该函数会在SDK初始化时调用进行设置，保证后续AppListInterface对该接口使用有效

#### virtual void AppListInterface::ShowPreviousUI ()[pure virtual]

通知切换到前一画面

##### 返回:

无

##### 参见:

##### 注解:

通知SDK需要切换到前一画面或App列表画面

* 1. rpcValueInterface
     1. 接口说明

rpc数据读取接口

* + 1. 成员函数说明

#### virtual bool rpcValueInterface::asBool () const[pure virtual]

将当前rpcValueInterface对象转换为bool类型

##### 返回:

rpcValueInterface数据值

##### 参见:

##### 注解:

#### virtual double rpcValueInterface::asDouble () const[pure virtual]

将当前rpcValueInterface对象转换为double类型

##### 返回:

rpcValueInterface数据值

##### 参见:

##### 注解:

#### virtual float rpcValueInterface::asFloat () const[pure virtual]

将当前rpcValueInterface对象转换为float类型

##### 返回:

rpcValueInterface数据值

##### 参见:

##### 注解:

#### virtual int rpcValueInterface::asInt () const[pure virtual]

将当前rpcValueInterface对象转换为int类型

##### 返回:

rpcValueInterface数据值

##### 参见:

##### 注解:

#### virtual std::string rpcValueInterface::asString () const[pure virtual]

将当前rpcValueInterface对象转换为std::string类型

##### 返回:

rpcValueInterface数据值

##### 参见:

##### 注解:

#### virtual unsigned int rpcValueInterface::asUInt () const[pure virtual]

将当前rpcValueInterface对象转换为unsigned int类型

##### 返回:

rpcValueInterface数据值

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isArray () const[pure virtual]

判断当前rpcValueInterface对象是否为数组类型

##### 返回:

为数组类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isBool () const[pure virtual]

判断当前rpcValueInterface对象是否为bool类型

##### 返回:

为bool类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isDouble () const[pure virtual]

判断当前rpcValueInterface对象是否为double类型

##### 返回:

为double类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isInt () const[pure virtual]

判断当前rpcValueInterface对象是否为int类型

##### 返回:

为int类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isInt64 () const[pure virtual]

判断当前rpcValueInterface对象是否为64位int类型

##### 返回:

为64位int类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isIntegral () const[pure virtual]

判断当前rpcValueInterface对象是否为整数类型

##### 返回:

为整数类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isMember (const char \* *key*) const[pure virtual]

判断是否存在某个字段

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *key* | 字段名，空字符结尾 |

##### 返回:

如果对象具有指定字段名的成员，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isNull () const[pure virtual]

判断当前rpcValueInterface对象是否为空类型

##### 返回:

为空类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isNumeric () const[pure virtual]

判断当前rpcValueInterface对象是否为数字类型

##### 返回:

为数字类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isObject () const[pure virtual]

判断当前rpcValueInterface对象是否为对象类型

##### 返回:

为对象类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isString () const[pure virtual]

判断当前rpcValueInterface对象是否为字符串类型

##### 返回:

为字符串类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isUInt () const[pure virtual]

判断当前rpcValueInterface对象是否为unsigned int类型

##### 返回:

为unsigned int类型，则返回true

##### 参见:

##### 注解:

#### virtual bool rpcValueInterface::isUInt64 () const[pure virtual]

判断当前rpcValueInterface对象是否为64位unsigned int类型

##### 返回:

为64位unsigned int类型，则返回true

##### 参见:

##### 注解:

#### virtual rpcValueInterface& rpcValueInterface::operator[] (int *index*)[pure virtual]

通过基于零的索引访问数组元素

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *index* | 文件名长度 |

##### 返回:

**rpcValueInterface**& rpc数据结构

##### 参见:

##### 注解:

在数组大小为索引+ 1，如果数组包含元素数小于索引，则会返回null

#### virtual const rpcValueInterface& rpcValueInterface::operator[] (int *index*) const[pure virtual]

通过基于零的索引访问数组元素

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *index* | 文件名长度 |

##### 返回:

**rpcValueInterface**& rpc数据结构

##### 参见:

##### 注解:

在数组大小为索引+ 1，如果数组包含元素数小于索引，则会返回null，返回值不能修改

#### virtual rpcValueInterface& rpcValueInterface::operator[] (const char \* *key*)[pure virtual]

通过字段关键字访问对象元素

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *key* | 字段关键字字符串 |

##### 返回:

**rpcValueInterface**& rpc数据结构

##### 参见:

##### 注解:

如果不存在，则创建空成员，关键字被限制为2^30-1个字符，超过该值将导致异常

#### virtual const rpcValueInterface& rpcValueInterface::operator[] (const char \* *key*) const[pure virtual]

通过字段关键字访问对象元素

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *key* | 字段关键字字符串 |

##### 返回:

**rpcValueInterface**& rpc数据结构

##### 参见:

##### 注解:

如果不存在，则创建空成员，关键字被限制为2^30-1个字符，超过该值将导致异常，返回值不能修改

#### virtual unsigned int rpcValueInterface::size () const[pure virtual]

获取对象或数组的元素个数

##### 返回:

对象或数组的元素个数

##### 参见:

##### 注解:

1. 调用侧实现的接口
   1. rpcValueInterface
      1. 接口说明

调用侧按照要求需要实现的接口，用于控制画面的显示部分

* + 1. 成员函数说明

#### virtual void UIInterface::initAppHMI ()[pure virtual]

初始化UI画面部分

##### 返回:

无

##### 参见:

##### 注解:

需要实现初始化HMI资源，UI部分各个画面实现类的创建及初始化，保证后续对画面的操作有效

#### virtual void UIInterface::onAppActive ()[pure virtual]

通知激活App

##### 返回:

无

##### 参见:

##### 注解:

App列表中点击App图标会触发该事件

#### virtual void UIInterface::onAppShow (int *type*)[pure virtual]

通知显示指定画面

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *type* | 画面ID |

##### 返回:

无

##### 参见:

**AppCommon.h** ShowType枚举值定义

##### 注解:

切换到指定画面实现类实例进行显示

#### virtual void UIInterface::onAppStop ()[pure virtual]

通知停止App

##### 返回:

无

##### 参见:

##### 注解:

退出App时会触发该事件

#### virtual void UIInterface::onAppUnregister (int *appId*)[pure virtual]

通知取消App注册

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *appId* | app对应的id |

##### 返回:

无

##### 参见:

##### 注解:

App取消注册时会触发该事件

#### virtual void UIInterface::OnEndAudioPassThru ()[pure virtual]

通知停止语音录音

##### 返回:

无

##### 参见:

##### 注解:

停止语音录音时会触发该事件

#### virtual void UIInterface::onVideoStreamStart ()[pure virtual]

通知开启视频流传输

##### 返回:

无

##### 参见:

##### 注解:

开启视频流传输时会触发该事件

#### virtual void UIInterface::onVideoStreamStop ()[pure virtual]

通知停止视频流传输

##### 返回:

无

##### 参见:

##### 注解:

停止视频流传输时会触发该事件

#### virtual void UIInterface::SetAppListInterface (AppListInterface \* *pList*)[pure virtual]

设置AppListInterface接口

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *pList* | AppListInterface接口 |

##### 返回:

无

##### 参见:

##### 注解:

该函数会在SDK初始化时调用进行设置，保证后续UIInterface对该接口使用有效

#### virtual void UIInterface::SetSDLStatus (bool *bConnect*)[pure virtual]

通知当前SDL的连接状态

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *bConnect* | 连接状态，是否连接 |

##### 返回:

无

##### 参见:

##### 注解:

#### virtual void UIInterface::ShowDeviceList ()[pure virtual]

通知刷新设备列表

##### 返回:

无

##### 参见:

##### 注解:

收到刷新设备列表通知时会触发该事件

#### virtual void UIInterface::tsSpeak (int *VRID*, std::string *strText*)[pure virtual]

通知tts语音播放

##### 参数:

|  |  |  |
| --- | --- | --- |
| in | *VRID* | 请求类型 |
| in | *strText* | 语音播放文本 |

##### 返回:

无

##### 参见:

**AppCommon.h** VRID枚举值定义

##### 注解:

实现TTS语音文本的转换并进行播放

1. 数据结构
   1. command
      1. 结构说明

菜单命令项信息

* + 1. 成员字段说明

#### int command::i\_appID

App ID

#### int command::i\_cmdID

cmd ID

#### unsigned char command::i\_ImageType

菜单命令项图标类型

#### int command::i\_menuID

菜单 ID

#### int command::i\_parentID

父菜单 ID

#### int command::i\_position

菜单命令项所在的位置

#### std::string command::str\_ImagePath

菜单命令项图标路径

#### std::string command::str\_menuName

菜单名

* 1. DeviceData
     1. 结构说明

设备信息

* + 1. 成员字段说明

#### std::string DeviceData::id

设备id

#### std::string DeviceData::name

设备名

1. 枚举类型说明
   1. enum ShowType

画面ID枚举类型

##### 枚举值:

|  |  |
| --- | --- |
| ID\_APPLINK | App列表 |
| ID\_DEVICEVIEW | 设备列表 |
| ID\_CHOICESET | ChoiceSet画面 |
| ID\_COMMAND | 菜单画面 |
| ID\_SHOW | Show画面 |
| ID\_ALERT | Alert画面 |
| ID\_AUDIOPASSTHRU | 录音画面 |
| ID\_CHOICESETVR | 保留 |
| ID\_SCROLLMSG | ScrollMessage画面 |
| ID\_SLIDER | Slider画面 |
| ID\_NOTIFY | 保留 |
| ID\_MEDIACLOCK | MediaClockTimer画面 |
| ID\_VIDEOSTREAM | VideoStream画面 |
| ID\_MAIN | Main画面 |
| ID\_UI\_MAX |  |

* 1. enum TOUCH\_TYPE

屏幕操作枚举类型

##### 枚举值:

|  |  |
| --- | --- |
| TOUCH\_START | 按下 |
| TOUCH\_MOVE | 移动 |
| TOUCH\_END | 弹起 |

1. 宏定义
   1. Alert画面执行状态

#define **ALERT\_TIMEOUT**  0 *超时*

#define **ALERT\_CLICK\_SOFTBUTTON**  1 *点击软按钮*

#define **ALERT\_ABORTED**  2 *中断*

* 1. ScrollMessage画面执行状态

#define **SCROLLMESSAGE\_TIMEOUT**  0 *超时*

#define **SCROLLMESSAGE\_CLICK\_SOFTBUTTON**  1 *点击软按钮*

#define **SCROLLMESSAGE\_REJECTED**  2 *拒绝*

* 1. 按钮按下模式

#define **BUTTON\_SHORT**  0 *短按*

#define **BUTTON\_LONG**  1 *长按*

* 1. ChoiceSet画面执行状态

#define **PERFORMINTERACTION\_TIMEOUT**  10 *超时*

#define **PERFORMINTERACTION\_CHOICE**  0 *选择VR命令*

* 1. Slider画面执行状态

#define **SLIDER\_OK**  0 *执行成功*

#define **SLIDER\_TIMEOUT**  10 *超时*

#define **SLIDER\_ABORTED**  5 *中断*

* 1. 录音画面执行状态

#define **PERFORMAUDIOPASSTHRU\_TIMEOUT**  0

#define **PERFORMAUDIOPASSTHRU\_RETYP**  7

#define **PERFORMAUDIOPASSTHRU\_DONE**  0

#define **PERFORMAUDIOPASSTHRU\_CANCEL**  5

* 1. TTS执行状态

#define **SPEEK\_OK**  0

#define **SPEEK\_INTERRUPTED**  5

1. 用法示例
   1. RPC数据读取接口用法示例
      1. 取Alert画面数据

原始rpc数据格式如下：

|  |  |
| --- | --- |
| 名字 | 请求内容 |
| UI.Alert | **{**  **"id"**: 49**,**  **"jsonrpc"**: "2.0"**,**  **"method"**: "UI.Alert"**,**  **"params"**: **{**  **"alertStrings"**: **[**  **{**  **"fieldName"**: "alertText1"**,**  **"fieldText"**: "Alert Line 1"  **},**  **{**  **"fieldName"**: "alertText2"**,**  **"fieldText"**: "Alert Line 2"  **},**  **{**  **"fieldName"**: "alertText3"**,**  **"fieldText"**: "Alert Line 3"  **}**  **],**  **"alertType"**: "BOTH"**,**  **"appID"**: 12077**,**  **"duration"**: 0**,**  **"progressIndicator"**: true**,**  **"softButtons"**: **[**  **{**  **"isHighlighted"**: true**,**  **"softButtonID"**: 5552**,**  **"systemAction"**: "DEFAULT\_ACTION"**,**  **"text"**: "Close"**,**  **"type"**: "TEXT"  **},**  **{**  **"isHighlighted"**: true**,**  **"softButtonID"**: 5553**,**  **"systemAction"**: "DEFAULT\_ACTION"**,**  **"text"**: "Close"**,**  **"type"**: "TEXT"  **},**  **{**  **"isHighlighted"**: true**,**  **"softButtonID"**: 5554**,**  **"systemAction"**: "DEFAULT\_ACTION"**,**  **"text"**: "Close"**,**  **"type"**: "TEXT"  **}**  **]**  **}**  **}** |

读取方法如下：

// 获取 Alert画面Rpc数据接口

rpcValueInterface& pObj = AppControl->*getAlertJson*();

// 判断alertStrings字段是否存在

if (pObj["params"].*isMember*("alertStrings")) {

// 获取alertStrings字段数组类型元素个数

iCount = pObj["params"]["alertStrings"].*size*() >

3 ? 3 : pObj["params"]["alertStrings"].*size*();

for (int i = 0; i < iCount; ++i) {

// 获取alertStrings每个元素中fieldText对应的数值

AppBase::SetEdlidedText(m\_aAlertLab+i,

pObj["params"]["alertStrings"][i]["fieldText"].*asString*().c\_str(),

width()\*0.9);

}

}

// 获取duration字段值并转换为整型

m\_Timer.start(pObj["params"]["duration"].*asInt*());

按照数据组织结构（参考文档《HMI SDK UI 设计指导说明书.docx》），一层一层获取数据值