HMISDK使用说明

# 简介

HMI 分三层结构：

1. 最底层连接层：负责与SDL七个通道的消息交互；
2. AppData数据管理层：将收到的Json消息解析、存储、控制；
3. UI层：由AppData层控制UI显示哪一界面；

# 三层结构

## 最底层连接层：

IMessageInterface

virtual void onRequest(Json::Value) = 0;

virtual void onNotification(Json::Value) = 0;

virtual void onResult(Json::Value) = 0;

virtual void onRawData(void \* p, int iLength) = 0;

virtual void onError(std::string error) = 0;

## AppData数据管理层接口

AppListInterface

//提供给UI当前在线的App列表；

virtual std::vector <Json::Value > getNewAppJsonVector() = 0;

//通知给连接层的三个消息：进入、离开、退出；

virtual void OnAppActivated(int appID) = 0;

virtual void OnApplicationOut(int appID) = 0;

virtual void OnApplicationExit(int appID) = 0;

//提供给UI进入某一App获取所有数据的接口类；

virtual AppDataInterface\* getAppDataInterface() = 0;

AppDataInterface

//提供给UI获取某一App的各UI界面的数据；

virtual Json::Value getShowData() = 0;

virtual std::vector<SMenuCommand> getCommandList() = 0;

virtual std::vector<SMenuCommand> getCommandList(int subMenuID) = 0;

virtual Json::Value getAlertJson() = 0;

virtual Json::Value getSlider() = 0;

virtual Json::Value getScrollableMsgJson() = 0;

virtual Json::Value getAudioPassThruJson() = 0;

virtual Json::Value getInteractionJson() = 0;

virtual std::string getUrlString() = 0;

//界面上所有响应事件需通知给连接层的消息；

virtual void OnSoftButtonClick(int sbID, int mode) = 0;

virtual void OnCommandClick(int appID,int cmdID) = 0;

virtual void OnAlertResponse(int alertID, int reason) = 0;

virtual void OnScrollMessageResponse(int smID, int reason) = 0;

virtual void OnSliderResponse( int code, int sliderid, int sliderPosition) = 0;

virtual void OnTTSSpeek(int speekID, int code) = 0;

virtual void OnPerformAudioPassThru(int appID, int performaudiopassthruID, int code) = 0;

virtual void OnPerformInteraction(int code, int performInteractionID, int choiceID) = 0;

virtual void OnMenuBtnClick(std::string btnText) = 0;

## UI层接口

UIInterface

virtual void onAppShow(int type) = 0;

virtual void onAppActivated(AppDataInterface\* pInterface) = 0;

virtual void onAppClose() = 0;

virtual void onAppRefresh() = 0;

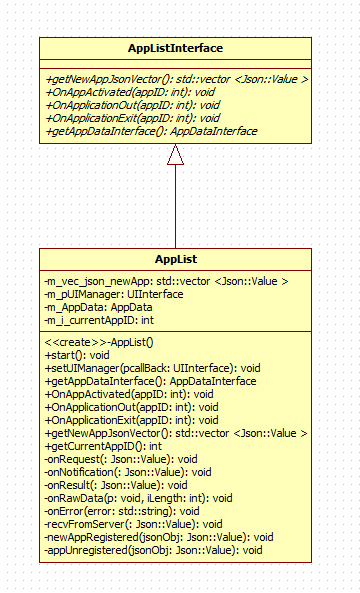
virtual void onTestVideoStreamStart() = 0;

virtual void onTestVideoStreamStop() = 0;

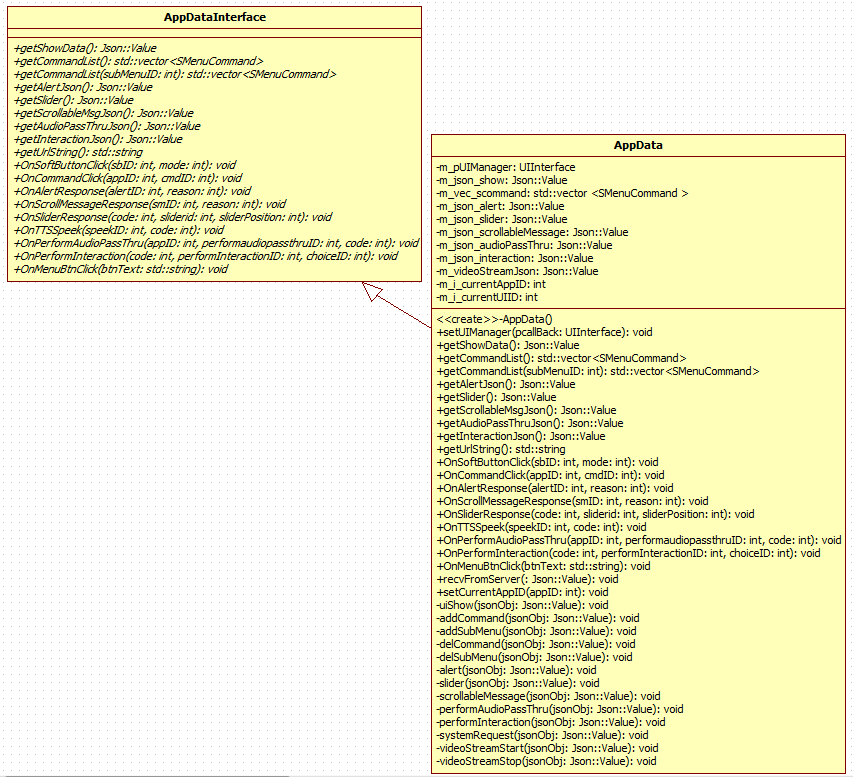
virtual void setMediaColckTimer(Json::Value jsonObj) = 0;

# AppData数据管理层对象

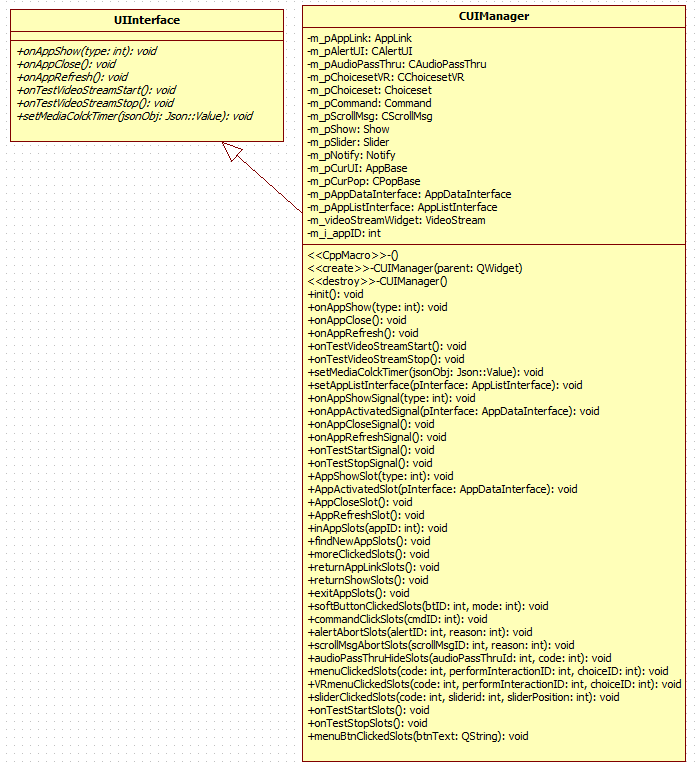
## AppList



## AppData



## UIManager



# 使用方法

在main() 函数里定义

AppList m\_appList;

CUIManager m\_ui;

m\_appList.setUIManager(&m\_ui);

m\_appList.start();

m\_ui.setAppListInterface(&m\_appList);

m\_ui.init();

## 在CUIManager 中实现相应接口函数

CUIManager类继承于UIInterface接口类，在CUIManager中实现onAppShow等相关接口函数。

onAppShow(int type);

type表明需要显示哪一个界面的ID。

## 在CUIManager 中声明对象

AppDataInterface\* m\_pAppDataInterface;

AppListInterface\* m\_pAppListInterface;

## 实现相关API函数

void CUIManager::init()

{

//显示App列表主界面

}

void CUIManager::setAppListInterface(AppListInterface\* pInterface)

{

m\_pAppListInterface = pInterface;

m\_pAppDataInterface = m\_pAppListInterface->*getAppDataInterface*();

}

其它界面的一些操作则需要用到对象m\_pAppListInterface 和m\_pAppDataInterface

## AppListInterface类对象使用

暂时划出App界面回到app列表主界面

m\_pAppListInterface->*OnApplicationOut*(m\_i\_appID);

进入App界面

m\_pAppListInterface->*OnAppActivated*(m\_i\_appID);

退出当前App界面

m\_pAppListInterface->*OnApplicationExit*(m\_i\_appID);

获取当面连接上的所有App的name和id

std::vector <Json::Value > pVecNewApp;

pVecNewApp = m\_pAppListInterface->*getNewAppJsonVector*();

//appName

pVecNewApp.at(i)["params"]["application"]["appName"].asString().data()

//appID

pVecNewApp.at(i)["params"]["application"]["appID"].asInt()

**获取到的appList中所有在线app中每一个app的Json数据例子：**

{

"jsonrpc" : "2.0",

"method" : "BasicCommunication.OnAppRegistered",

"params" : {

"application" : {

"appID" : 846930886,

"appName" : "Sina Weibo",

"appType" : [ "NAVIGATION" ],

"deviceName" : "10.27.0.132",

"hmiDisplayLanguageDesired" : "EN-US",

"icon" : "",

"isMediaApplication" : true,

"ngnMediaScreenAppName" : "Sina "

},

"vrSynonyms" : [ "Sina Weibo" ]

}

}

## AppDataInterface类对象使用

m\_pAppDataInterface->*OnSoftButtonClick*(btID, mode);

// mode: SHORT

or LONG

m\_pAppDataInterface->*OnCommandClick*(m\_i\_appID, cmdID);

m\_pAppDataInterface->*OnAlertResponse*(alertID, reason);

//reason: ALERT\_TIMEOUT

or ALERT\_CLICK\_SOFTBUTTON

or ALERT\_ABORTED

m\_pAppDataInterface->*OnScrollMessageResponse*(smID, reason);

//reason: SCROLLMESSAGE\_TIMEOUT

or SCROLLMESSAGE\_CLICK\_SOFTBUTTON

or SCROLLMESSAGE\_REJECTED

m\_pAppDataInterface->*OnPerformAudioPassThru*(m\_i\_appID, audioPassThruId, code);

//code: PERFORMAUDIOPASSTHRU\_TIMEOUT

or PERFORMAUDIOPASSTHRU\_RETYP

or PERFORMAUDIOPASSTHRU\_DONE

Or PERFORMAUDIOPASSTHRU\_CANCEL

m\_pAppDataInterface->*OnPerformInteraction*(code, performInteractionID, choiceID);

//code: PERFORMINTERACTION\_TIMEOUT

Or PERFORMINTERACTION\_CHOICE

m\_pAppDataInterface->*OnSliderResponse*(code, sliderid,sliderPosition);

//code: SLIDER\_OK

or SLIDER\_TIMEOUT

or SLIDER\_ABORTED

开始录音

m\_pAppDataInterface->*OnVRStartRecord*();

结束录音

m\_pAppDataInterface->*OnVRCancelRecord*();

**获取各界面的Json数据**

Json::Value *getShowData*();

Json::Value *getAlertJson*();

Json::Value *getSlider*();

Json::Value *getScrollableMsgJson*();

Json::Value *getAudioPassThruJson*();

Json::Value *getInteractionJson*();

**获取其它数据**

std::string *getUrlString*();

std::vector<SMenuCommand> *getCommandList*();

std::vector<SMenuCommand> *getCommandList*(int subMenuID);