FAW HS7

IVI System Design

Audio-API User Guide

Released on the: 2017.05.15

Version Number: 0.600

Neusoft Automotives ChangChun

**Version History**

| No. | Version | Section | Brief Description | Date | Rseponsible |
| --- | --- | --- | --- | --- | --- |
| 1 | 0.500 | All | Initial Create | 2016-10-08 | WangQi |
| 2 | 0.600 | 3.3 | Update API | 2017-05-15 | WangQi |
| 3 | 0.700 | 3.2 | Update API | 2017-12-1 | WangQi |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |
|  |  |  |  |  |  |

Content

[1 概述 4](#_Toc483229493)

[2 ·相关文件 4](#_Toc483229494)

[3 接口说明 4](#_Toc483229495)

[3.1 Audio-API使用约束 4](#_Toc483229496)

[3.2 数据定义 5](#_Toc483229497)

[3.2.1 声音焦点Focus 5](#_Toc483229498)

[3.2.2 音频设备audioDevice 5](#_Toc483229499)

[3.2.3 语音流类型streamType 5](#_Toc483229500)

[3.3 AppAudioStream基类定义 7](#_Toc483229501)

[3.3.1 创建语音流接口 8](#_Toc483229502)

[3.3.2 获取streamID接口 8](#_Toc483229503)

[3.3.3 获取streamID字符串接口 8](#_Toc483229504)

[3.3.4 申请AudioFocus接口 8](#_Toc483229505)

[3.3.5 释放AudioFocus接口 9](#_Toc483229506)

[3.3.6 设置静音接口 9](#_Toc483229507)

[3.3.7 声音焦点变化通知 9](#_Toc483229508)

[4 Refrences 10](#_Toc483229509)

# 概述

本文档主要描述Audio-API。Audio-API主要提供两个方面接口。

* AudioFocus管理：包括申请，释放，以及Focus变化通知。
* 提示音发声管理：包括按键音，雷达音，提示音的发声接口。

# ·相关文件

头文件：/usr/include/appSDK/ama\_audioTypes.h //audio相关类型定义

/usr/include/appSDK/AppAudioStream.h //AppAudioStream基类定义

库文件：/usr/lib/libappSDK.so

# 接口说明

## Audio-API使用约束

约束条件：

* 使用Audio-API前需要首先初始化AppSDK；
* 应用程序需使用Audio-API分配的StreamID创建PulseAudio语音流；

## 数据定义

数据类型定义在：/usr/include/appSDK/ama\_audioTypes.h

### 声音焦点Focus

Focus状态定义如下：

enum ama\_app\_focusState\_e{

E\_FOUCS\_LOSE = 0x01,

E\_FOUCS\_LOSE\_TRANSIENT = 0x02,

E\_FOUCS\_GAIN = 0x10

};

* E\_FOUCS\_LOSE：失去声音焦点；
* E\_FOUCS\_LOSE\_TRANSIENT：暂时失去声音焦点；
* E\_FOUCS\_GAIN：获取声音焦点；

说明：在HS7系统中，只有多媒体类型音频会有暂时失去声音焦点的情况。

典型应用场景如下：

1. 系统开机后默认进行FM播放；
2. 用户进入music应用，播放歌曲；此时，music应用申请声音焦点，并获得声音焦点进行播放；
3. 蓝牙电话呼入：由于蓝牙应用申请声音焦点并获得声音焦点，music应用会收到暂时失去焦点的通知，music应用收到此通知时暂停音乐播放；
4. 蓝牙通话结束：由于蓝牙应用放弃声音焦点，music应用会再次收到获取声音焦点的通知，music收到此通知时恢复音乐播放；

### 音频设备audioDevice

音频设备audioDevice定义如下：

enum ama\_audioDevice\_e{

E\_IVI\_MIC = 0x0,

E\_IVI\_SPEAKER = 0x1,

E\_RSE\_L\_HEADSET = 0x2,

E\_RSE\_R\_HEADSET = 0x3,

E\_INVALID\_DEVICE = 0xf

};

* E\_IVI\_MIC：麦克设备
* E\_IVI\_SPEAKER：主扬声器
* E\_RSE\_L\_HEADSET：左后娱乐系统耳机
* E\_RSE\_R\_HEADSET：右后娱乐系统耳机

### 语音流类型streamType

系统语音流类型定义如下：

enum ama\_streamType\_e{

//focusType = RADAR volumeType=frontRadar

E\_STREAM\_FL\_RADAR = 0x001, //The audio Stream for radar Front Left

E\_STREAM\_FR\_RADAR = 0x002, //The audio Stream for radar Front Right

//focusType = RADAR volumeType=RearRadar

E\_STREAM\_RL\_RADAR = 0x011, //The audio Stream for radar Rear Left

E\_STREAM\_RR\_RADAR = 0x012, //The audio Stream for radar Rear Right

//focusType = NOTIFICATION volumeType=NOTIFICATION

E\_STREAM\_NOTIFICATION = 0x121, //the audio Stream for notifications

E\_STREAM\_BACKDOOR\_NOTI = 0x122, //the audio Stream for RR backdoor notifications

E\_STREAM\_FEEDBACK\_NOTI = 0x123, //the audio Stream for feedbackTone

//focusType = SOFTKEY volumeType=SOFTKEY

E\_STREAM\_SOFTKEY = 0x231, //the audio Stream for soft key

//focusType = MEDIA volumeType=MEDIA

E\_STREAM\_TUNER = 0x341, //the audio Stream for TUNER

E\_STREAM\_MUSIC = 0x342, //the audio Stream for (EMMC,USB) music playback

E\_STREAM\_VIDEO\_AUDIO = 0x343, //the audio Stream for sound of video playback

E\_STREAM\_BT\_AUDIO = 0x344, //the audio Stream for sound of bt music

E\_STREAM\_PHONELINK\_AUDIO = 0x345, //the audio Stream for sound of phonelink

E\_STREAM\_3THAPP\_AUDIO = 0x346, //the audio Stream for sound of 3th app

E\_STREAM\_FM = 0x347, //the audio stream for fm tuner

E\_STREAM\_AM = 0x348, //the audio stream for am tuner

E\_STREAM\_IPOD\_AUDIO = 0x349, //the audio stream for IPOD

E\_STREAM\_CARPLAY\_AUDIO = 0x34A, //the audio stream for carplay

E\_STREAM\_CARLIFE\_AUDIO = 0x34B, //the audio stream for carlife

//focusType = SMSTTS volumeType=SMSTTS

E\_STREAM\_SMS\_TTS = 0x451, //the audio Stream for TTS of SMS

//focusType = VRTTS volumeType=VRTTS

E\_STREAM\_VR\_TTS = 0x561, //the audio Stream for TTS of VR

//focusType = NAVITTS volumeType=NAVITTS

E\_STREAM\_NAVI\_TTS = 0x671, //the audio Stream for TTS of navi

//focusType = Phone volumeType=Phone

E\_STREAM\_CALL\_RING = 0x781, //the audio Stream of call ring

E\_STREAM\_BT\_CALL = 0x782, //the audio Stream for bt call

//focusType = B&I-CALL volumeType=Phone

E\_STREAM\_B\_CALL = 0x881, //The audio Stream for B\_Call

E\_STREAM\_I\_CALL = 0x882, //The audio Stream for I\_Call

//focusType = E-Call volumeType=E-Call

E\_STREAM\_E\_CALL = 0x991, //The audio Stream for E\_Call

//focusType = RVC-Mode volumeType=unknown

E\_STREAM\_RVC\_MODE = 0xAF1, //It's not a audio stream,it is control audio state by RVC-Mode

//focusType = MIC volumeType=unknown

E\_STREAM\_PHONE\_MIC = 0xEF1, //the audio Stream for phone mic

E\_STREAM\_VR\_MIC = 0xEF2, //the audio Stream for VR mic

E\_STREAM\_B\_CALL\_MIC = 0xEF3, //the audio Stream for B\_CALL mic

E\_STREAM\_I\_CALL\_MIC = 0xEF4, //the audio Stream for I\_CALL mic

E\_STREAM\_E\_CALL\_MIC = 0xEF5, //the audio Stream for E\_CALL mic

E\_STREAM\_LOCAL\_VR\_MIC = 0xEF6 //the audio Stream for E\_CALL mic

};

## AppAudioStream基类定义

AppAudioStream基类定义在：/usr/include/appSDK/AppAudioStream.h

AppAudioStream定义如下：

class AppAudioStream : public enable\_shared\_from\_this<AppAudioStream>

{

public:

AppAudioStream(ama\_streamType\_t streamType);

AppAudioStream(ama\_seatID\_t seatID,ama\_appID\_t appID,ama\_streamType\_t streamType);

virtual ~AppAudioStream();

ama\_streamID\_t streamID(void);

string streamIDStr(void);

//fullname:requestAudioFocusForDefaultDevice

ama\_Error\_e requestAudioFocusForDefDev(void);

//fullname:requestAudioFocusForDevice

ama\_Error\_e requestAudioFocusForDev(ama\_audioDeviceType\_t audioDevice);

//fullname:abandonAudioFocusForDefaultDevice

ama\_Error\_e abandonAudioFocusForDefDev(void);

//fullname:abandonAudioFocusForDevice

ama\_Error\_e abandonAudioFocusForDev(ama\_audioDeviceType\_t audioDevice);

ama\_Error\_e setStreamMuteState(bool isMute);//for all device

ama\_Error\_e duckAudio(int durationsms,int volume);

ama\_Error\_e unduckAudio(int durationsms);

protected:

friend class AppAudioServiceImpl;

//fullname:releaseAudioFocusForDevice

virtual void OnAudioFocusChangedForDefDev(ama\_focusSta\_t newFocusSta);

virtual void OnAudioFocusChanged(ama\_audioDeviceType\_t audioDevice,ama\_focusSta\_t newFocusSta);

private:

void\* pAudioServiceImpl;

ama\_streamID\_t mStreamID;

};

应用程序若想发声需要创建语音流类继承于AppAudioStream基类，调用基类中的方法申请或者放弃Focus焦点，同时必须实现OnAudioFocusChangedForDefDev ()虚函数，获得Focus变化通知。

### 创建语音流接口

|  |  |  |
| --- | --- | --- |
| *declare* | new xxxStream(ama\_streamTypeID\_t streamType); | |
| *description* | 创建语音流的构造函数 | |
| *parameter* | type ama\_streamType\_t | streamType语音流类型，3.2.3中说明 |
| *return* | null | |

示例：new MusicStream(E\_STREAM\_MUSIC);

### 获取streamID接口

|  |  |  |
| --- | --- | --- |
| *declare* | ama\_streamID\_t streamID(void) | |
| *description* | 获取streamID的值 | |
| *parameter* | type void | null |
| *return* | streamID的值 | |

### 获取streamID字符串接口

|  |  |  |
| --- | --- | --- |
| *declare* | string streamIDStr(void) | |
| *description* | 获取streamID字符串 | |
| *parameter* | type void | null |
| *return* | streamID的字符串表示 | |

应用程序应调用此接口获取streamID的字符串并创建语音流，因为PulseAudio需要字符串参数。

此接口至关重要，语音流之所以能够发声是AppManger会将获取AudioFocus的语音流连接至对应的音频设备。而连接时，需要知道语音流的标识进行识别。此streamID作为此标识必须赋给语音流！

### 申请AudioFocus接口

|  |  |  |
| --- | --- | --- |
| *declare* | ama\_Error\_e requestAudioFocusForDefDev (void); | |
| *description* | 申请默认音频设备AudioFocus | |
| *parameter* | type void | null |
| *return* | 0：操作成功 负数：错误码（错误码定义在/usr/include/appSDK/ama\_types.h） | |

|  |  |  |
| --- | --- | --- |
| *declare* | ama\_Error\_e requestAudioFocusForDev(ama\_audioDevice\_t audioDevice); | |
| *description* | 申请制定音频设备AudioFocus | |
| *parameter* | ama\_audioDevice\_t | 音频设备类型，在3.2.2中说明 |
| *return* | 0：操作成功 负数：错误码（错误码定义在/usr/include/appSDK/ama\_types.h） | |

应用程序必须有发声需求才可调用此接口（即获取AudioFocus后立即输出声音），确保有发声需求的应用能够及时获取到AudioFocus。

此处有两个申请声音焦点的接口：

* 申请默认音频设备接口，其默认对应关系如下：
  + 前屏应用对应申请主扬声器设备；
  + 左后屏应用对应申请左后屏耳机；
  + 右后屏应用对应申请右后屏耳机；‘
* 申请指定音频设备接口，此接口有两种情况使用：
  + 申请mic设备的focus状态，需要指定音频设备；
  + 不按默认对应关系情况，例如前屏应用在音频同步功能下向右后屏输出时；

### 释放AudioFocus接口

|  |  |  |
| --- | --- | --- |
| *declare* | ama\_Error\_e abandonAudioFocusForDefDev(void); | |
| *description* | 释放默认音频设备AudioFocus | |
| *parameter* | type void | null |
| *return* | 0：操作成功 负数：错误码（错误码定义在/usr/include/appSDK/ama\_types.h） | |

|  |  |  |
| --- | --- | --- |
| *declare* | ama\_Error\_e abandonAudioFocusForDev(ama\_audioDevice\_t audioDevice); | |
| *description* | 释放指定音频设备AudioFocus | |
| *parameter* | ama\_audioDevice\_t | 音频设备类型，在3.2.2中说明 |
| *return* | 0：操作成功 负数：错误码（错误码定义在/usr/include/appSDK/ama\_types.h） | |

应用程序声音输出完毕后立即调用此接口，确保有发声需求的应用能够及时获取到AudioFocus。

此处两个接口使用同3.3.4.

### 设置静音接口

|  |  |  |
| --- | --- | --- |
| *declare* | ama\_Error\_e setStreamMuteState(bool isMute) | |
| *description* | 设置语音流静音 | |
| *parameter* | type bool | true:设置为静音 false：解除静音 |
| *return* | 0：操作成功 负数：错误码（错误码定义在/usr/include/appSDK/ama\_types.h） | |

### 声音焦点变化通知

|  |  |  |
| --- | --- | --- |
| *declare* | virtual void OnAudioFocusChangedForDefDev(ama\_focusSta\_t newFocusSta); | |
| *description* | 默认音频设备焦点变化处理函数 | |
| *parameter* | ama\_focusSta\_t | 焦点状态类型，3.2.1中说明 |
| *return* | null | |

|  |  |  |
| --- | --- | --- |
| *declare* | virtual void OnAudioFocusChanged(ama\_audioDevice\_t audioDevice,ama\_focusSta\_t newFocusSta); | |
| *description* | 指定音频设备焦点变化处理函数 | |
| *parameter* | ama\_audioDevice\_t | 音频设备类型，在3.2.2中说明 |
| *parameter* | ama\_focusSta\_t | 焦点状态类型，在3.2.1中说明 |
| *return* | null | |

应用程序创建语音流类必须实现此方法，并针对不同的焦点状态做对应处理。

* 获得声音焦点：播放音频
* 失去声音焦点：必须调用释放AudioFocus接口。并按照式样暂停或者静音。
* 暂时失去声音焦点：不能调用释放AudioFocus接口，按照式样要求暂停或者静音。等待再次获取声音焦点，再正常播放

例如：启动IVI，进入Music应用，点击播放按键，Music申请声音焦点，获取声音焦点后，播放。

进入Video应用，播放视频。Video申请声音焦点，Video获取声音焦点，播放。

此时Music失去声音焦点，释放声音焦点，并进入暂停状态。

有蓝牙电话呼入，蓝牙电话申请并获得声音焦点。播放蓝牙电话声音。

此时，Video暂时失去声音焦点，进入暂停状态。

蓝牙通话结束后，蓝牙电话放弃声音焦点。

此时，Video再次获得声音焦点，继续播放。

以上场景覆盖了三种状态变化的处理流程。

关于AudioFocus的使用说明：

在应用程序调用申请AudioFocus方法后，会立即收到一个AudioFocus获取或者失去的通知。在此后的过程中会一直收到AudioFocus变化的通知，直至应用程序调用释放AudioFocus方法后。

# Refrences