Desay API Description

|  |  |  |  |
| --- | --- | --- | --- |
| **Version** | **Date** | **Document Owner** | **Changes** |
| 1.0 | 2-28 | PanJuan | 初稿audio, MIC, CarProperty |
| 1.1 | 03-04 | PanJuan | 修改导航播报获取焦点的STREAM类型 |
| 1.2 | 03-06 | XieJun | 新增获取车身信息，发送多媒体信息等接口。 |
| 1.3 | 03-12 | XieJun | 新增车身信息回调接口，图像EVS接口等。 |
| 1.4 | 03-27 | XieJun | 调整接口，更新文档 |
| 1.5 | 04-10 | XieJun | 增加BaseManager，增加AudioManager接口，增加enhanceMemory接口 |
| 1.6 | 04-17 | XieJun | 1.增加获取AudioFocus信息接口；  2.增加蓝牙接口；  3.增加EV接口；  4.EVS增加doneWithFrame接口 |

目录

[1 android.media.AudioManager.java 3](#_Toc38030569)

[1.1 requestAudioFocus(AudioFocusRequest focusRequest) 3](#_Toc38030570)

[1.2 abandonAudioFocusRequest(AudioFocusRequest focusRequest） 4](#_Toc38030571)

[1.3 setStreamVolume (int streamType, int index, int flags) 4](#_Toc38030572)

[2 Mic抢占 4](#_Toc38030573)

[3 com.ford.base.FordBaseManager.java 4](#_Toc38030574)

[4 com.ford.car.FordCarInfoManager.java 5](#_Toc38030575)

[4.1 public static FordCarInfoManager getInstance(Context context) 5](#_Toc38030576)

[4.2 int getCarModel() 5](#_Toc38030577)

[4.3 String getVinCode() 5](#_Toc38030578)

[4.4 String getCarColor() 5](#_Toc38030579)

[4.5 String getESN() 5](#_Toc38030580)

[4.6 String getPCBANumber() 6](#_Toc38030581)

[4.7 String getAssemblyNumber() 6](#_Toc38030582)

[4.8 String getVMCUVersion() 6](#_Toc38030583)

[4.9 String getSerialNo() 6](#_Toc38030584)

[4.10 String getSystemModel() 6](#_Toc38030585)

[4.11 String getSystemVersion() 6](#_Toc38030586)

[4.12 int getCarPowerType() 6](#_Toc38030587)

[4.13 void setActiveStatus(int status) 6](#_Toc38030588)

[4.14 int getActiveStatus() 6](#_Toc38030589)

[4.15 String getSystemType() 6](#_Toc38030590)

[4.16 String getCarType() 7](#_Toc38030591)

[4.17 String getCarStyle() 7](#_Toc38030592)

[4.18 boolean has360Camera() 7](#_Toc38030593)

[5 com.ford.car.FordCarVendorExtensionManager.java 7](#_Toc38030594)

[5.1 void startServiceAsUser(Intent intent) 7](#_Toc38030595)

[5.2 String getBootAnimIsStopped() 7](#_Toc38030596)

[5.3 void setNeedLoginCheck(String loginCheck) 7](#_Toc38030597)

[5.4 String getNeedLoginCheck() 7](#_Toc38030598)

[5.5 void setAuthResult(int authed) 7](#_Toc38030599)

[6 com.ford.car.FordCarSensorManager.java 8](#_Toc38030600)

[6.1 public static FordCarSensorManager getInstance(Context context) 8](#_Toc38030601)

[6.2 int getCarSpeedUnit() 8](#_Toc38030602)

[6.3 float getCarSpeed() 8](#_Toc38030603)

[6.4 int getCarDriveMode() 8](#_Toc38030604)

[6.5 int getCarDoorOpenStatus(int doorPos) 8](#_Toc38030605)

[6.6 float getCarEnduranceMileage() 9](#_Toc38030606)

[6.7 float getCarEndurancePercent() 9](#_Toc38030607)

[6.8 int getCarEnduranceWarnStatus() 9](#_Toc38030608)

[6.9 float getCarFuelPercent() 9](#_Toc38030609)

[6.10 int getCarFuelWarnStatus() 9](#_Toc38030610)

[6.11 int getCarGear() 9](#_Toc38030611)

[6.12 int getCarMileageUnit() 9](#_Toc38030612)

[6.13 float getCarTirePressure(int tirePos) 10](#_Toc38030613)

[6.14 int getCarTirePressureUnit() 10](#_Toc38030614)

[6.15 float getCarTireTemperature(int tirePos) 10](#_Toc38030615)

[6.16 int getCarTireTemperatureUnit() 10](#_Toc38030616)

[6.17 int getCarTireStatus(int tirePos) 10](#_Toc38030617)

[6.18 int getCarTireSupportResult() 11](#_Toc38030618)

[6.19 float getCarTotalMileage() 11](#_Toc38030619)

[6.20 int getCarWindowOpenStatus(int windowPos) 11](#_Toc38030620)

[6.21 addListener(SensorChangeListener listener) 11](#_Toc38030621)

[6.22 removeListener(SensorChangeListener listener) 11](#_Toc38030622)

[6.23 clearListeners() 11](#_Toc38030623)

[6.24 回调接口 11](#_Toc38030624)

[6.24.1 驾驶模式 12](#_Toc38030625)

[6.24.2 车速 12](#_Toc38030626)

[6.24.3 点火状态 12](#_Toc38030627)

[6.24.4 里程 12](#_Toc38030628)

[6.24.5 里程单位 12](#_Toc38030629)

[6.24.6 油量 13](#_Toc38030630)

[6.24.7 档位 13](#_Toc38030631)

[6.24.8 车窗 13](#_Toc38030632)

[6.24.9 车门 13](#_Toc38030633)

[6.25 sendNaviToTCU(Bundle bundle) 13](#_Toc38030634)

[7 com.ford.car.FordCarHvacManager.java 14](#_Toc38030635)

[7.1 public static FordCarHvacManager getInstance(Context context) 14](#_Toc38030636)

[7.2 boolean isCarFrontClimateOpen() 14](#_Toc38030637)

[7.3 boolean isCarRearClimateOpen() 14](#_Toc38030638)

[7.4 boolean isSupportAAR() 15](#_Toc38030639)

[7.5 int getCarClimateCycleMode() 15](#_Toc38030640)

[7.6 void setCarClimateCycleMode(int mode) 15](#_Toc38030641)

[7.7 void setCarFrontClimateOnOff(boolean isOpen) 15](#_Toc38030642)

[7.8 void setCarRearClimateOnOff(boolean isOpen) 15](#_Toc38030643)

[7.9 addListener(ClimateChangeListener listener) 15](#_Toc38030644)

[7.10 removeListener(ClimateChangeListener listener) 16](#_Toc38030645)

[7.11 clearListeners() 16](#_Toc38030646)

[8 com.ford.media.MediaEventManager 16](#_Toc38030647)

[8.1 USB广播 16](#_Toc38030648)

[8.2 public static FordMediaEventManager getInstance(Context context) 17](#_Toc38030649)

[8.3 void sendVRTips(String text) 17](#_Toc38030650)

[8.4 void sendID3Info(ID3Info id3Info) 17](#_Toc38030651)

[8.5 void sendPlayState(int state) 17](#_Toc38030652)

[8.6 void sendPlayMode(int mode) 17](#_Toc38030653)

[8.7 addListener(ActionListener listener) 17](#_Toc38030654)

[8.8 removeListener(ActionListener listener) 17](#_Toc38030655)

[8.9 clearListeners() 18](#_Toc38030656)

[8.10 回调接口 18](#_Toc38030657)

[9 com.ford.camera.EVSCameraManager 18](#_Toc38030658)

[10 com.ford.vendor.FordEnhanceMemoryManager 18](#_Toc38030659)

[11 com.ford.vendor.FordEVChargeManager 18](#_Toc38030660)

[12 com.ford.vendor.FordBluetoothManager 18](#_Toc38030661)

# android.media.AudioManager.java

## requestAudioFocus(AudioFocusRequest focusRequest)

Android原生接口，用于申请焦点，需要构造一个audioAttributes，source类型通过audioAttributes的usage传递给系统。

具体实现可以遵循原生的接口使用：https://developer.android.google.cn/reference/android/media/AudioManager?hl=en#requestAudioFocus(android.media.AudioFocusRequest)

其中，导航播报的参数如下：

CONTENT\_TYPE: com.ford.audio.FordAudioManager.STREAM\_NAVIGATION；

正常播报USAGE:  AudioAttributes.USAGE\_ASSISTANCE\_NAVIGATION\_GUIDANCE；

导航复播USAGE: AudioAttributes.USAGE\_ASSISTANCE\_ACCESSIBILITY。

注：导航复播时，焦点优先级最高，比语音都高

## abandonAudioFocusRequest(AudioFocusRequest focusRequest）

abandonAudioFocus的audioAttributes参数需要和其requestAudioFocus时传入的attribute对应  
**特别重要：**request时的attribute和abandon时的attribute必须为同一个attribute。

## setStreamVolume (int streamType, int index, int flags)

Android原生接口，调节音量。

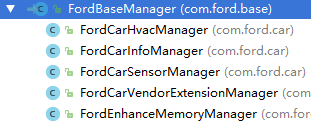
其中，导航音量传入的参数streamtype对应com.ford.audio.FordAudioManager.STREAM\_NAVIGATION。

# Mic抢占

1. VR APP自己定义一个public static final int HOTWORD = 1999;
2. Manifest.xml加上权限”android.permission.CAPTURE\_AUDIO\_HOTWORD”
3. 语音app在收到其他apps抢占mic的消息后，需要将降噪设置设回正常模式，以免影响其他app的mic使用。

# com.ford.base.FordBaseManager.java

BaseManager为各个Manager基类，提供connect和disconnect接口。



继承自BaseManager的管理类，调用方式都是xxxxManager.getInstance().connect(ServiceConnection)，收到onServiceConnected回调后，表示Service连接成功，可以开始业务逻辑。

# com.ford.car.FordCarInfoManager.java

## public static FordCarInfoManager getInstance(Context context)

获取单例。

## int getCarModel()

获取当前车型。

返回值说明：

**public static final int *CAR\_MODEL\_CD542H*** = 0;*//CD542-High***public static final int *CAR\_MODEL\_CD542M*** = 1;*//CD542-Middle***public static final int *CAR\_MODEL\_CD542L*** = 2;*//CD542-Low***public static final int *CAR\_MODEL\_CX727H*** = 3;*//CX727-High***public static final int *CAR\_MODEL\_CX727L*** = 4;*//CX727-Low***public static final int *CAR\_MODEL\_U725H*** = 5;*//U725-High***public static final int *CAR\_MODEL\_U725L*** = 6;*//U725-Low***public static final int *CAR\_MODEL\_P702*** = 7;*//P702***public static final int *CAR\_MODEL\_U554H*** = 8;*//U554-High***public static final int *CAR\_MODEL\_U554L*** = 9;*//U554-Low***public static final int *CAR\_MODEL\_U625H*** = 10;*//U625-High***public static final int *CAR\_MODEL\_U625M*** = 11;*//U625-Middle***public static final int *CAR\_MODEL\_U625L*** = 12;*//U625-Low***public static final int *CAR\_MODEL\_CD764*** = 13;*//CD764*

## String getVinCode()

获取Vin码。

## String getCarColor()

获取车辆颜色

返回值：

**public static final** String ***CAR\_COLOR\_BLACK*** = **"black"**;  
**public static final** String ***CAR\_COLOR\_WHITE*** = **"white"**;  
**public static final** String ***CAR\_COLOR\_RED*** = **"red"**;  
**public static final** String ***CAR\_COLOR\_SILVER*** = **"silver"**;

## String getESN()

获取车机设备电子序列号。

## String getPCBANumber()

获取车机PCBA零件号。

## String getAssemblyNumber()

获取车机设备型号。

## String getVMCUVersion()

获取VMCU版本号。

## String getSerialNo()

获取序列号，等效于SystemProperties.*get*(**"ro.serialno"**);

## String getSystemModel()

获取系统信号，等效于SystemProperties.*get*(**"ro.product.model"**);

## String getSystemVersion()

获取系统版本，等效于SystemProperties.*get*(**"ro.build.id"**);

## int getCarPowerType()

获取车辆动力类型

返回值，0：油，1：电，2：BEV

**public static final int *POWER\_TYPE\_OIL*** = 0;//油  
**public static final int *POWER\_TYPE\_ELE*** = 1;//电  
**public static final int *POWER\_TYPE\_BEV*** = 2;//BEV

## void setActiveStatus(int status)

写激活状态。

## int getActiveStatus()

读取激活状态。

## String getSystemType()

获取ROM版本类型，debug版本返回“debug”，release版本返回“release”。

## String getCarType()

获取车辆样式，返回值范围：*"4DR", "SUV", "5DR"*

## String getCarStyle()

获取车辆款式，返回值范围：*"DI","DJ","DP","DF","DG","DM","DH","DL","DX","DR","MP","DQ","DN","DM","DJ","DK","GV","TP","DD","HZ","JY"*

## boolean has360Camera()

是否有360Camera。true表示有360，false表示没有。

# com.ford.car.FordCarVendorExtensionManager.java

## void startServiceAsUser(Intent intent)

对外提供startServiceAsUser接口：



## String getBootAnimIsStopped()

开机动画是否停止。返回“stopped”表示已停止。

## void setNeedLoginCheck(String loginCheck)

设置是否强制检验账号登录。"logincheckclose"表示免登录，"logincheckopen"表示需要强制登录。

## String getNeedLoginCheck()

获取是否强制检验账号登录。"logincheckclose"表示免登录，"logincheckopen"表示需要强制登录。

## void setAuthResult(int authed)

保存账号授权协议的结果。1：已授权，0：未授权。

# com.ford.car.FordCarSensorManager.java

## public static FordCarSensorManager getInstance(Context context)

获取Manager单例。

## int getCarSpeedUnit()

获取车速单位。

返回值说明：

**public static final int *SPEED\_UNIT\_CMS*** = 0;*// "cm/s";***public static final int *SPEED\_UNIT\_MS*** = 1;*//"m/s";***public static final int *SPEED\_UNIT\_KMH*** = 2;*//"km/h";*

## float getCarSpeed()

获取车速。

## int getCarDriveMode()

获取驾驶模式

返回值说明：

**public static final int *DRIVE\_MODE\_SOLO*** = 0;*//solo mode***public static final int *DRIVE\_MODE\_CO\_PILOT*** = 1;*//co-pilot mode***public static final int *DRIVE\_MODE\_INDIVIDUAL*** = 2;*//individual mode*

## int getCarDoorOpenStatus(int doorPos)

获取各位置的车门打开状态。

参数：

**public static final int *DOOR\_FRONT\_LEFT*** = 0; //左前车门  
**public static final int *DOOR\_FRONT\_RIGHT*** = 1; //右前车门  
**public static final int *DOOR\_REAR\_LEFT*** = 2; //左后车门  
**public static final int *DOOR\_REAR\_RIGHT*** = 3; //右后车门  
**public static final int *DOOR\_OUTSIDE\_TRUNK*** = 4; //后备箱外车门  
**public static final int *DOOR\_INSIDE\_TRUNK*** = 5; //后备箱内车门

返回值，1：关闭，2：打开

**public static final int *WINDOW\_DOOR\_STATUS\_CLOSE*** = 1;  
**public static final int *WINDOW\_DOOR\_STATUS\_OPEN*** = 2;

## float getCarEnduranceMileage()

获取车辆续航里程

## float getCarEndurancePercent()

获取车辆续航百分比

## int getCarEnduranceWarnStatus()

获取续航警告状态

返回值，0：正常，1：告警

**public static final int *ENDURANCE\_STATUS\_OK*** = 0; **public static final int *ENDURANCE\_STATUS\_WARN*** = 1;

## float getCarFuelPercent()

获取剩余油量百分比。比如返回值为78.1，代表还剩余78.1%的油量。

## int getCarFuelWarnStatus()

获取油量告警状态

返回值，0：正常，1：告警

**public static final int *FUEL\_STATUS\_OK*** = 0; *//fuel status is ok***public static final int *FUEL\_STATUS\_WARN*** = 1; *//fuel status is warning*

## int getCarGear()

获取车辆档位

返回值，0：N档，1：P档，2：R档，3：D档

**public static final int *GEAR\_NEUTRAL*** = 0; *//N***public static final int *GEAR\_PARKING*** = 1; *//P***public static final int *GEAR\_REVERSE*** = 2; *//R***public static final int *GEAR\_DRIVE*** = 3; *//D*

## int getCarMileageUnit()

获取里程单位

返回值，

**public static final int *MILEAGE\_UNIT\_MILE*** = 0;*//"英里"***public static final int *MILEAGE\_UNIT\_KM*** = 1;*//"公里"*

## float getCarTirePressure(int tirePos)

获取胎压

参数：

**public static final int *TIRE\_FRONT\_LEFT*** = 0;//前左轮胎  
**public static final int *TIRE\_FRONT\_RIGHT*** = 1;//前右轮胎  
**public static final int *TIRE\_REAR\_LEFT*** = 2;//后左轮胎  
**public static final int *TIRE\_REAR\_RIGHT*** = 3;//后右轮胎

## int getCarTirePressureUnit()

获取胎压单位

返回值，

**public static final int *TIRE\_PRESSURE\_UNIT\_BAR*** = 0;*//bar***public static final int *TIRE\_PRESSURE\_UNIT\_KPA*** = 1;*//kpa*

## float getCarTireTemperature(int tirePos)

获取胎温

参数：

**public static final int *TIRE\_FRONT\_LEFT*** = 0;//前左轮胎  
**public static final int *TIRE\_FRONT\_RIGHT*** = 1;//前右轮胎  
**public static final int *TIRE\_REAR\_LEFT*** = 2;//后左轮胎  
**public static final int *TIRE\_REAR\_RIGHT*** = 3;//后右轮胎

## int getCarTireTemperatureUnit()

获取胎温单位

返回值，

**public static final int *TIRE\_TEMPERATURE\_UNIT\_CEN*** = 0;*//centigrade*

## int getCarTireStatus(int tirePos)

获取轮胎状态

参数：

**public static final int *TIRE\_FRONT\_LEFT*** = 0;//前左轮胎  
**public static final int *TIRE\_FRONT\_RIGHT*** = 1;//前右轮胎  
**public static final int *TIRE\_REAR\_LEFT*** = 2;//后左轮胎  
**public static final int *TIRE\_REAR\_RIGHT*** = 3;//后右轮胎

返回值：

**public static final int *TIRE\_STATUS\_NORMAL*** = 1; //正常状态  
**public static final int *TIRE\_STATUS\_UNNORMAL*** = 2; //异常状态

## int getCarTireSupportResult()

车辆是否支持获取胎温、胎压

返回值，

**public static final int *SUPPORT\_TIRE\_NO*** = 0; *//都不支持***public static final int *SUPPORT\_TIRE\_TEMP\_PRESSURE*** = 1; *//都支持***public static final int *SUPPORT\_TIRE\_PRESSURE*** = 2; *//仅支持获取胎压*

## float getCarTotalMileage()

获取车辆总里程

## int getCarWindowOpenStatus(int windowPos)

获取车窗打开状态

参数，

**public static final int *WINDOW\_POS\_DRIVER*** = 0; //驾驶位车窗  
**public static final int *WINDOW\_POS\_FRONT\_PASSENGER*** = 1; //副驾车窗  
**public static final int *WINDOW\_POS\_REAR\_DRIVER*** = 2; //驾驶位后排车窗  
**public static final int *WINDOW\_POS\_REAR\_PASSENGER*** = 3; //副驾位后排车窗

返回值，1：关闭，2：打开

**public static final int *WINDOW\_DOOR\_STATUS\_CLOSE*** = 1;  
**public static final int *WINDOW\_DOOR\_STATUS\_OPEN*** = 2;

## addListener(SensorChangeListener listener)

增加listener。

## removeListener(SensorChangeListener listener)

移除listener。

## clearListeners()

清空listener集合。

## 回调接口

**public interface** SensorChangeListener {  
  
 **void** onSensorChange(**int** eventType, **int**[] intValues, **float**[] floatValues);  
}

FordCarSensorManager的各种传感器类信息，例如车速、档位等变化，统一回调onSensorChange接口。eventType表示对应事件类型，intValues保存回调的int型值，floatValues保存回调的float型值。

例如，收到onSensorChange(0, intValues, floatValues)，eventType为0，代表驾驶模式发生了变化，驾驶模式值是一个int型，所以取intValues[0]即可；没有float值，floatValues为null。

### 驾驶模式

eventType:**public static final int *EVENT\_TYPE\_DRIVE\_MODE*** = 0;

回调值：

intValues[0]，代表驾驶模式的值，值定义见4.4节。

### 车速

eventType:**public static final int *EVENT\_TYPE\_SPEED\_VALUE*** = 1;

回调值：

floatValues[0]，代表车速值

### 点火状态

eventType:**public static final int *EVENT\_TYPE\_IGNITION*** = 2;

回调值：

intValues[0]，代表点火状态值，值定义：

**public static final int *IGNITION\_STATUS\_OFF*** = 0;*//ignition-off***public static final int *IGNITION\_STATUS\_ON*** = 1;*//ignition-on*

### 里程

eventType:**public static final int *EVENT\_TYPE\_MILEAGE\_VALUE*** = 3;

回调值：

floatValues[0]，代表当前的续航里程；floatValues[1]，代表总里程。

### 里程单位

eventType:**public static final int *EVENT\_TYPE\_MILEAGE\_UNIT*** = 4;

回调值：

intValues[0]，代表里程单位，值定义见4.12节。

### 油量

eventType:**public static final int *EVENT\_TYPE\_FUEL*** = 5;

回调值：

intValues[0]，代表油量告警状态，值定义见4.11节。

floatValues[0]，代表剩余油量百分比。

### 档位

eventType:**public static final int *EVENT\_TYPE\_GEAR*** = 6;

回调值：

intValues[0]，代表档位，值定义见4.13节。

### 车窗

eventType:**public static final int *EVENT\_TYPE\_WINDOW*** = 7;

回调值：

intValues[0]，代表车窗位置，值定义见4.20节。

intValues[1]，代表车窗开关状态，值定义见4.20节。

### 车门

eventType:**public static final int *EVENT\_TYPE\_DOOR*** = 8;

回调值：

intValues[0]，代表车门位置，值定义见4.5节。

intValues[1]，代表车门开关状态，值定义见4.5节。

## sendNaviToTCU(Bundle bundle)

将处理后的导航信息，发送给车机系统。

Bundle参数，用法与原先保持一致：



# com.ford.car.FordCarHvacManager.java

## public static FordCarHvacManager getInstance(Context context)

获取Manager单例。

## boolean isCarFrontClimateOpen()

前排空调是否打开。

返回值，true：打开，false：关闭

## boolean isCarRearClimateOpen()

后排空调是否打开。

返回值，true：打开，false：关闭

## boolean isSupportAAR()

是否有AAR功能。

返回值，true：有，false：无

## int getCarClimateCycleMode()

获取空调循环模式。

返回值，0：内循环，1：外循环

**public static final int *CLIMATE\_CYCLE\_MODE\_INNER*** = 0;  
**public static final int *CLIMATE\_CYCLE\_MODE\_OUTER*** = 1;

## void setCarClimateCycleMode(int mode)

设置空调循环模式。

参数，0：内循环，1：外循环

**public static final int *CLIMATE\_CYCLE\_MODE\_INNER*** = 0;  
**public static final int *CLIMATE\_CYCLE\_MODE\_OUTER*** = 1;

## void setCarFrontClimateOnOff(boolean isOpen)

设置前排空调开关。

参数，true：打开空调，false：关闭空调

## void setCarRearClimateOnOff(boolean isOpen)

设置后排空调开关。

参数，true：打开空调，false：关闭空调

## addListener(ClimateChangeListener listener)

设置空调状态变化的监听器。

**public interface** ClimateChangeListener {  
  
 */\*\*  
 \* Called when the climate's open status has changed.  
 \** ***@param postion*** *which climate's open status has changed.  
 \* 0: the front climate  
 \* 1: the rear climate  
 \** ***@param isOpen*** *climate's open status value.  
 \* 0: open  
 \* 1: close  
 \** ***@param operation*** *How did the user turn the climate on/off  
 \* 0: use app on the screen, default value.  
 \* 1: use hardKey on the car.  
 \*/* **void** onOpenStatusChange(**int** postion, **boolean** isOpen, **int** operation);  
  
 */\*\*  
 \* Called when the climate's cycle mode has changed.  
 \** ***@param mode*** *value after mode change  
 \*/* **void** onCycleModeChange(**int** mode);  
  
 */\*\*  
 \* Called when the climate's change has error.  
 \** ***@param errorCode*** *error code  
 \* 100: the front climate open error  
 \* 101: the front climate close error  
 \* 110: the rear climate open error  
 \* 111: the rear climate close error  
 \* 200: the climate cycle mode change error  
 \*/* **void** onError(**int** errorCode);  
}

## removeListener(ClimateChangeListener listener)

移除添加的listener。

## clearListeners()

清空listener集合。

# com.ford.media.MediaEventManager

## USB广播

增加了USB扫描失败、设备不支持的广播Action，其余的USB插入、拔出、扫描等，使用原生广播Action即可。用法与原生保持一致。

*/\*\*  
 \* Broadcast Action: The media scanner has failed to scan a directory.  
 \* The path to the failure scanned directory is contained in the Intent.mData field.  
 \*/***public static final** String ***ACTION\_MEDIA\_SCANNER\_SCAN\_FAILED*** = **"ford.intent.action.MEDIA\_SCANNER\_SCAN\_FAILED"**;  
  
*/\*\*  
 \* Broadcast Action: The device is not supported  
 \* The path to the mount point for the unsupport device is contained in the Intent.mData field.  
 \*/***public static final** String ***ACTION\_MEDIA\_UNSUPPORT*** = **"ford.intent.action.MEDIA\_UNSUPPORT"**;

## public static FordMediaEventManager getInstance(Context context)

获取单例

## void sendVRTips(String text)

发送VR提示文本给车机底部导航栏显示。

## void sendID3Info(ID3Info id3Info)

将随心听播放的ID3信息，发送给仪表显示。ID3Info的字段定义，与百度《随心听交互协议\_3.8.pdf》一致。

## void sendPlayState(int state)

将随心听播放状态，发送给仪表显示。state定义，

**public static final int *PLAY\_STATE\_BUFFERING*** = 1; *//buffering***public static final int *PLAY\_STATE\_PLAYING*** = 2; *//play***public static final int *PLAY\_STATE\_PAUSE*** = 3; *//pause***public static final int *PLAY\_STATE\_STOP*** = 4; *//stop*

## void sendPlayMode(int mode)

将随心听播放模式，发送给仪表显示。mode定义：

**public static final int *PLAY\_MODE\_SINGLE\_CYCLE*** = 0; *//single cycle***public static final int *PLAY\_MODE\_SHUFFLE\_PLAY*** = 1; *//shuffle play***public static final int *PLAY\_MODE\_LIST\_LOOP*** = 2; *//list loop***public static final int *PLAY\_MODE\_ORDER\_PLAY*** = 3; *//order play*

## addListener(ActionListener listener)

增加listener。

## removeListener(ActionListener listener)

移除listener。

## clearListeners()

清空listener。

## 回调接口

**public interface** ActionListener {  
 **void** onAction(**int** action);  
}

当用户通过仪表控制多媒体时，会回调onAction方法，action参数定义：

**public static final int *ACTION\_PLAY*** = 0;*//播放***public static final int *ACTION\_PAUSE*** = 1;*//暂停***public static final int *ACTION\_PREVIOUS*** = 2;*//上一曲***public static final int *ACTION\_NEXT*** = 3;*//下一曲*

# com.ford.camera.EVSCameraManager

图像evs接口，请阅读《evc\_camera-api\_description-v2.1-20200303172110-kkovev.pdf》接口文档。

# com.ford.vendor.FordEnhanceMemoryManager

em接口，按之前邮件excel表格定义的，请参考邮件。

# com.ford.vendor.FordEVChargeManager

EV接口，按之前邮件excel表格定义的，请参考邮件。

# com.ford.vendor.FordBluetoothManager

提供监听蓝牙状态的接口，蓝牙的地址、通话状态等，均从回调接口的SVCallInfo中取出即可。

**void** addListener(IBtHfpListener listener)

*/\*\*  
 \* 通话状态变化回调  
 \*/***public interface** IBtHfpListener {  
 **void** onPhoneCallStateChange(SVCallInfo svCallInfo);  
}

通话状态：

**static class** CallState {  
 **public static final int *CALL\_STATE\_ACTIVE*** = 0;  
 **public static final int *CALL\_STATE\_HELD*** = 1;  
 **public static final int *CALL\_STATE\_DIALING*** = 2;  
 **public static final int *CALL\_STATE\_ALERTING*** = 3;  
 **public static final int *CALL\_STATE\_INCOMING*** = 4;  
 **public static final int *CALL\_STATE\_WAITING*** = 5;  
 **public static final int *CALL\_STATE\_HELD\_BY\_RESPONSE\_AND\_HOLD*** = 6;  
 **public static final int *CALL\_STATE\_TERMINATED*** = 7;  
 **public static final int *CALL\_STATE\_MISSCALL*** = 8;  
}