**AmyRobot Android SDK manual**

**AmyRobot Android SDK manual**

**目录**

**Contents**

[1. 集成SDK 4](#_Toc31632)

[1. Integrate SDK 4](#_Toc31632)

[2. 开启远程控制服务 5](#_Toc30080)

[2. Open Remote Control Service 5](#_Toc30080)

[3. 获取地图列表 7](#_Toc1794)

[3. Get Map List 7](#_Toc1794)

[4. 开启导航 7](#_Toc31355)

[4. Open Navigation 7](#_Toc31355)

[5. 停止导航 11](#_Toc30277)

[5. Stop Navigation 11](#_Toc30277)

[6. 获取导航状态 13](#_Toc21939)

[6. Get Navigation Status 13](#_Toc21939)

[7. 导航到标记界面 15](#_Toc32437)

[7. Navigate to Mark Interface 15](#_Toc32437)

[8. 获取标记列表 15](#_Toc11966)

[8. Get Mark List 15](#_Toc11966)

[9. 导航到标记点 17](#_Toc23210)

[9. Navigate to Marker 17](#_Toc23210)

[10. 取消导航 19](#_Toc11073)

[10. Cancel Navigation 19](#_Toc11073)

[11. 机器人控制API 21](#_Toc25373)

[11. Robot Control API 21](#_Toc25373)

[12. 导航控制API 23](#_Toc2705)

[12. Navigation Control API 23](#_Toc2705)

[13. 系统Action 24](#_Toc843)

[13. System Action 24](#_Toc843)

[14. 打开/关闭唤醒的方法(只适用2.0版本) 26](#_Toc30543)

[14. Method of Open/Close Wakeup (version 2.0 only) 26](#_Toc30543)

[15. 监听唤醒broadcast 27](#_Toc15961)

[15. Listen to Wakeup Broadcast 27](#_Toc15961)

[16. 第三方开发语音app时注意事项 27](#_Toc1770)

[16. Considerations When Developing a Voice App by a Third Party 27](#_Toc1770)

[17. AmyRobot A1 OS v1.0(已停止维护)启用/禁用系统语音的方法 28](#_Toc32597)

[17. AmyRobot A1 OS v1.0 (no longer maintened) Method to Enable/Disable System Voice 28](#_Toc32597)

[18. 电量广播及获得电量的方法 29](#_Toc24111)

[18. Power Broadcast & Method of Getting Battery Capacity 29](#_Toc24111)

版本修改记录：

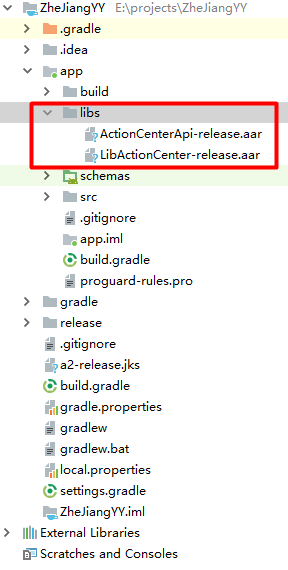
Version Modification Record:

|  |  |  |  |
| --- | --- | --- | --- |
| 版本  Version | 修改说明  Modification | 修改日期  Modification Date | 修改人  Modified by |
| V1.0  V1.0 | 新建Amy导航SDK使用说明文档，包括开启导航，停止导航，取消导航，获取地图列表，获取标记列表，导航到标记点等；  Create new instructions document for Amy navigation SDK, including open navigation, stop navigation, cancel navigation, get map list, getting mark list, navigating to marker, etc. | 2018-06-21  2018-06-21 | Mudeyu  mudeyu |
| V2.0  V2.0 | 添加机器人控制API，导航控制API；  Add robot control API, navigation control API; | 2018-08-02  2018-08-02 | Mudeyu  mudeyu |
| V3.0  V3.0 | 添加系统Action，包括音频，视频，语音播报，语音识别，视频呼叫等；  Add system Action, including audio, video, voice broadcast, voice recognition, video call, etc. | 2018-08-30  2018-08-30 | Mudeyu  mudeyu |
| V3.1  V3.1 | 修改集成sdk方法；调整语音播报api参数；调整语音识别api参数  Modify integrated sdk method; adjust voice broadcast API parameters; adjust voice recognition API parameters | 2018-11-28  2018-11-28 | Lizz  Lizz |
| V3.2  V3.2 | 加入打开/关闭唤醒的方法  Add open/close wakeup method | 2018-12-10  2018-12-10 | Lizz  Lizz |
| V3.3  V3.3 | 监听唤醒broadcast  Monitor wakeup broadcast  1.0版本 启用/禁用系统语音的方法  Method of enable/disable system voice, version 1.0 | 2019-01-23  2019-01-23 | Lizz  Lizz |
| V3.4  V3.4 | 删除语音相关说明  Delete voice related instructions  第三方开发语音app时注意事项  Considerations when developing a voice app by a third party | 2019-02-18  2019-02-18 | Lizz  lizz |
| V3.5  V3.5 | 加入电量广播及获取电量的方法  Method of add power broadcast and get power | 2019-03-06  2019-03-06 | Lizz  Lizz |

## 集成SDK Integrate SDK

1. 将库文件aar拷贝到libs 目录下

Copy library file aar to libs directory



1. build.gradle 导入aar

Build.gradle import aar



1. AndroidManifest.xml配置如下，其中**your.company.url**请修改为您所在公司的域名

AndroidManifest.xml is configured as follows, where **your.company.url should be** modified to the domain name of your company.



1. 在App-->attachBaseContext方法中添加如下初始化代码，其中**your.company.url**与AndroidManifest.xml中设定的一致

4) Add the following initialization code in the App-->attachBaseContext method, where **your.company.url** is the same as that set in AndroidManifest.xml



## 开启远程控制服务 Open Remote Control Service

先启动远程控制服务，确认版本号大于等于 1.10.2；

Start the remote control service first, and confirm that the version number is 1.10.2 or above;

查看版本：远程控制->更多设置->版本

View Version: Remote Control -> More Settings -> Version



打开SDKsamples/NavDemo/AmySDKProject进入主界面；如下图所示：

Open SDKsamples/NavDemo/AmySDKProject to enter the main interface; as shown below:



## 获取地图列表 3. Get Map List

点击刷新地图列表获取地图列表；

Click Refresh Map List to get map list;

代码示例：

Code example:

robotNavActionApi.getMapList(new ActionResultCallback<MapListEntity>() {

robotNavActionApi.getMapList(new ActionResultCallback<MapListEntity>() {

@Override

@Override

public void onActionSuccess(final MapListEntity mapListEntity) {

public void onActionSuccess(final MapListEntity mapListEntity) { // mapListEntity 地图列表

// mapListEntity map list

}

}

@Override

@Override

public void onActionFailed(int errCode, String errInfo, Throwable throwable) {

public void onActionFailed(int errCode, String errInfo, Throwable throwable) {

showToast("获取地图列表错误 " + errCode + ", " + errInfo);

showToast("get map list error" + errCode + ", " + errInfo);

}

}

});

});

## 开启导航 4. Open Navigation

点击开启默认导航或者点击地图，开启导航；

Click Open Default Navigation or click Map to open the navigation;

使用默认导航地图开启导航，需要先设置默认导航地图；

To use the default navigation map to open navigation, you need to set the default navigation map first;

开启成功返回 ActionCode.ACTION\_OK = 0；

Open success return ActionCode.ACTION\_OK = 0;

开启失败返回错误码：

Open failed return error code:

//ok

//ok

public static final int CODE\_SUCCESS = ActionCode.ACTION\_OK;

public static final int CODE\_SUCCESS = ActionCode.ACTION\_OK;

//错误

//error

public static final int CODE\_ERROR = 1000;

public static final int CODE\_ERROR = 1000;

//无效参数

// invalid parameters

public static final int CODE\_INVALID\_PARAMS = 1001;

public static final int CODE\_INVALID\_PARAMS = 1001;

//开启导航失败，一般是工控失败

// open navigation failed, generally IPC failure

public static final int CODE\_ERROR\_START\_NAV\_FAILED = 2000;

public static final int CODE\_ERROR\_START\_NAV\_FAILED = 2000;

//无效默认地图id错误，用户没有设置默认导航地图导致

//Invalid default map id error, as the user did not set the default navigation map

public static final int CODE\_ERROR\_INVALID\_DEFAULT\_MAP\_ID = 2001;

public static final int CODE\_ERROR\_INVALID\_DEFAULT\_MAP\_ID = 2001;

//该地图没有标记点，无法导航

//The map has no markers and cannot navigate

public static final int CODE\_ERROR\_MAP\_MARK\_POINTS\_IS\_EMPTY = 2002;

public static final int CODE\_ERROR\_MAP\_MARK\_POINTS\_IS\_EMPTY = 2002;

//标记列表中没有该标记点，无法导航

// There is no such marker in the mark list, can not navigate

public static final int CODE\_ERROR\_MAP\_MARK\_POINT\_NOT\_EXIST = 2003;

public static final int CODE\_ERROR\_MAP\_MARK\_POINT\_NOT\_EXIST = 2003;

//导航失败

//Navigation failed

public static final int CODE\_ERROR\_NAV\_FAILED = 2004;

public static final int CODE\_ERROR\_NAV\_FAILED = 2004;

//导航放弃

// Navigation give up

public static final int CODE\_ERROR\_NAV\_GIVEUP = 2005;

public static final int CODE\_ERROR\_NAV\_GIVEUP = 2005;

//导航丢失

//Navigation lost

public static final int CODE\_ERROR\_NAV\_LOST = 2006;

public static final int CODE\_ERROR\_NAV\_LOST = 2006;

//导航超时

// Navigation timeout

public static final int CODE\_ERROR\_NAV\_TIME\_OUT = 2007;

public static final int CODE\_ERROR\_NAV\_TIME\_OUT = 2007;

//导航取消

//Navigation cancel

public static final int CODE\_ERROR\_NAV\_CANCEL = 2008;

public static final int CODE\_ERROR\_NAV\_CANCEL = 2008;

//导航停止

//Navigation stop

public static final int CODE\_ERROR\_NAV\_STOP = 2009;

public static final int CODE\_ERROR\_NAV\_STOP = 2009;

**开启导航代码示例：**

**Example of open navigation code:**

int ret = robotNavActionApi.startNav(createStartNavCallBack());

int ret = robotNavActionApi.startNav(createStartNavCallBack());

if(ret == ActionCode.ACTION\_OK) {

if(ret == ActionCode.ACTION\_OK) {

showLoading("正在开启导航...");

showLoading("opening navigation...");

} else {

} else {

showToast("错误 " + ret);

showToast("error" + ret);

}

}

private ActionNotifyCallBack createStartNavCallBack() {

private ActionNotifyCallBack createStartNavCallBack() {

startNavCallBack = new ActionNotifyCallBack() {

startNavCallBack = new ActionNotifyCallBack() {

@Override

@Override

public void onRecvActionNotify(ActionRequest notifyRequest,

public void onRecvActionNotify(ActionRequest notifyRequest,

String srcURI, String notifyAction, final int notifyCode,

String srcURI, String notifyAction, final int notifyCode,

final String notifyInfo, String notifyParams) {

final String notifyInfo, String notifyParams) {

LogUtils.d(TAG, "onRecvActionNotify srcURI=" + srcURI + ", notifyAction=" + notifyAction

LogUtils.d(TAG, "onRecvActionNotify srcURI=" + srcURI + ", notifyAction=" + notifyAction

+ ", notifyCode=" + notifyCode + ", notifyInfo=" + notifyInfo

+ ", notifyCode=" + notifyCode + ", notifyInfo=" + notifyInfo

+ ", notifyParams=" + notifyParams);

+ ", notifyParams=" + notifyParams);

if (notifyCode == ActionCode.ACTION\_OK) {

if (notifyCode == ActionCode.ACTION\_OK) {

//成功

//success

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

showToast("开启导航成功");

showToast ("open navigation successful");

hideLoading();

hideLoading();

startActivity(new Intent(getActivity(), NavMarkPointListActivity.class));

startActivity(new Intent(getActivity(), NavMarkPointListActivity.class));

}

}

});

});

} else {

} else {

//错误，错误码：notifyCode, 错误信息：notifyInfo

//Error, error code: notifyCode, error information: notifyInfo

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

showToast("开启导航失败 " + notifyCode + ", " + notifyInfo);

showToast("open navigation failed" + notifyCode + ", " + notifyInfo);

}

}

});

});

}

}

}

}

};

};

return startNavCallBack;

return startNavCallBack;

}

}

## 停止导航 Stop Navigation

点击停止导航，执行停止导航；

Click Stop Navigation and execute stop navigation.

**代码示例：**

**Code example:**

robotNavActionApi.stopNav(new ActionNotifyCallBack() {

robotNavActionApi.stopNav(new ActionNotifyCallBack() {

@Override

@Override

public void onRecvActionNotify(ActionRequest notifyRequest,

public void onRecvActionNotify(ActionRequest notifyRequest,

String srcURI, String notifyAction, final int notifyCode,

String srcURI, String notifyAction, final int notifyCode,

final String notifyInfo, String notifyParams) {

final String notifyInfo, String notifyParams) {

LogUtils.d(TAG, "onRecvActionNotify srcURI=" + srcURI + ", notifyAction=" + notifyAction

LogUtils.d(TAG, "onRecvActionNotify srcURI=" + srcURI + ", notifyAction=" + notifyAction

+ ", notifyCode=" + notifyCode + ", notifyInfo=" + notifyInfo

+ ", notifyCode=" + notifyCode + ", notifyInfo=" + notifyInfo

+ ", notifyParams=" + notifyParams);

+ ", notifyParams=" + notifyParams);

if (notifyCode == ActionCode.ACTION\_OK) {

if (notifyCode == ActionCode.ACTION\_OK) {

//成功

//success

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

hideLoading();

hideLoading();

showToast("停止导航成功");

showToast ("stop navigation successful");

}

}

});

});

} else {

} else {

//错误，错误码：notifyCode, 错误信息：notifyInfo

//Error, error code: notifyCode, error information: notifyInfo

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

hideLoading();

hideLoading();

showToast("停止导航失败 " + notifyCode + ", " + notifyInfo);

showToast("stop navigation failed" + notifyCode + ", " + notifyInfo);

}

}

});

});

}

}

}

}

});

});

## 获取导航状态 6. Get Navigation Status

private void getNavState() {

private void getNavState() {

ViewUtils.setText(tvNavState, "正在获取导航状态");

ViewUtils.setText(tvNavState, "Getting navigation status");

robotNavActionApi.getNavState(new ActionNotifyCallBack() {

robotNavActionApi.getNavState(new ActionNotifyCallBack() {

@Override

@Override

public void onRecvActionNotify(ActionRequest notifyRequest,

public void onRecvActionNotify(ActionRequest notifyRequest,

String srcURI, String notifyAction, final int notifyCode,

String srcURI, String notifyAction, final int notifyCode,

final String notifyInfo, final String notifyParams) {

final String notifyInfo, final String notifyParams) {

if (notifyCode == ActionCode.ACTION\_OK) {

if (notifyCode == ActionCode.ACTION\_OK) {

//成功, notifyParams

//success, notifyParams

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

ViewUtils.setText(tvNavState, notifyParams);

ViewUtils.setText(tvNavState, notifyParams);

}

}

});

});

} else {

} else {

//错误，错误码：notifyCode, 错误信息：notifyInfo

//Error, error code: notifyCode, error information: notifyInfo

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

ViewUtils.setText(tvNavState, "err=" + notifyCode + ", " + notifyInfo);

ViewUtils.setText(tvNavState, "err=" + notifyCode + ", " + notifyInfo);

}

}

});

});

}

}

}

}

});

});

}

}

## 导航到标记界面 7. Navigate to Mark Interface

开启导航，自动跳转到导航到标记界面

Open navigation and auto jump to Navigation to Mark Interface



## 获取标记列表 8. Get Mark List

点击获取标记列表，获取地图对应的标记列表；

Click Get Mark List to get a mark list corresponding to the map;

可以选择根据标记坐标导航，根据标记名称导航；

You can choose to navigate based on mark coordinates or navigate based on mark name;

代码示例：

Code example:

private void getMarkPointList() {

private void getMarkPointList() {

robotNavActionApi.getMarkPointList(null, new ActionResultCallback<List<MarkPointEntity>>() {

robotNavActionApi.getMarkPointList(null, new ActionResultCallback<List<MarkPointEntity>>() {

@Override

@Override

public void onActionSuccess(final List<MarkPointEntity> markPointEntities) {

public void onActionSuccess(final List<MarkPointEntity> markPointEntities) {

LogUtils.e(TAG, "getMarkPointList ok");

LogUtils.e(TAG, "getMarkPointList ok");

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

if (isFinishing()) {

if (isFinishing()) {

return;

return;

}

}

showToast("获取标记列表成功");

showToast("get mark list successful");

onGetMarkPointList(markPointEntities);

onGetMarkPointList(markPointEntities);

}

}

});

});

}

}

@Override

@Override

public void onActionFailed(int errCode, String errInfo, Throwable throwable) {

public void onActionFailed(int errCode, String errInfo, Throwable throwable) {

LogUtils.e(TAG, "getMarkPointList failed errCode=" + errCode + ", errInfo=" + errInfo + ", " + throwable);

LogUtils.e(TAG, "getMarkPointList failed errCode=" + errCode + ", errInfo=" + errInfo + ", " + throwable);

showToast("获取标记列表失败 " + errCode + ", " + errInfo);

showToast("get mark list failed" + errCode + ", " + errInfo);

}

}

});

});

}

}

## 导航到标记点 9. Navigate to Marker

private void navToPoint(final MarkPointEntity markPointEntity) {

private void navToPoint(final MarkPointEntity markPointEntity) {

tvState.setText("正在导航到: " + markPointEntity.text);

tvState.setText("Navigating to: " + markPointEntity.text);

if (navToPointCallBack != null) {

if (navToPointCallBack != null) {

//取消前次监听

// cancel previous monitor

navToPointCallBack.cancel();

navToPointCallBack.cancel();

}

}

navToPointCallBack = new ActionNotifyCallBack() {

navToPointCallBack = new ActionNotifyCallBack() {

@Override

@Override

public void onRecvActionNotify(ActionRequest notifyRequest,

public void onRecvActionNotify(ActionRequest notifyRequest,

String srcURI, String notifyAction, final int notifyCode,

String srcURI, String notifyAction, final int notifyCode,

final String notifyInfo, final String notifyParams) {

final String notifyInfo, final String notifyParams) {

if (notifyCode == ActionCode.ACTION\_OK) {

if (notifyCode == ActionCode.ACTION\_OK) {

//成功, notifyParams

//success, notifyParams

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

String info = "导航到达: " + markPointEntity.text;

String info = "Navigation arrive: " + markPointEntity.text;

showToast(info);

showToast(info);

tvState.setText(info);

tvState.setText(info);

}

}

});

});

} else {

} else {

//错误，错误码：notifyCode, 错误信息：notifyInfo

//Error, error code: notifyCode, error information: notifyInfo

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

String info = "导航错误 " + notifyCode + ", " + notifyInfo;

String info = "navigation error" + notifyCode + ", " + notifyInfo;

showToast(info);

showToast(info);

tvState.setText(info);

tvState.setText(info);

}

}

});

});

}

}

}

}

};

};

LogUtils.d(TAG, "navToPoint isNavPointByPos =" + isNavPointByPos + ", markPointEntity=" + JSON.toJSONString(markPointEntity));

LogUtils.d(TAG, "navToPoint isNavPointByPos =" + isNavPointByPos + ", markPointEntity=" + JSON.toJSONString(markPointEntity));

if (isNavPointByPos) {

if (isNavPointByPos) {

robotNavActionApi.navToPointByPostion(markPointEntity.realX, markPointEntity.realY, markPointEntity.realAngle, navToPointCallBack);

robotNavActionApi.navToPointByPostion(markPointEntity.realX, markPointEntity.realY, markPointEntity.realAngle, navToPointCallBack);

} else {

} else {

robotNavActionApi.navToPointByName(markPointEntity.text, navToPointCallBack);

robotNavActionApi.navToPointByName(markPointEntity.text, navToPointCallBack);

}

}

}

}

## 取消导航 10. Cancel Navigation

代码示例：

Code example:

private void cancelNav() {

private void cancelNav() {

robotNavActionApi.cancelNav(new ActionNotifyCallBack() {

robotNavActionApi.cancelNav(new ActionNotifyCallBack() {

@Override

@Override

public void onRecvActionNotify(ActionRequest notifyRequest,

public void onRecvActionNotify(ActionRequest notifyRequest,

String srcURI, String notifyAction, final int notifyCode,

String srcURI, String notifyAction, final int notifyCode,

final String notifyInfo, String notifyParams) {

final String notifyInfo, String notifyParams) {

LogUtils.d(TAG, "onRecvActionNotify srcURI=" + srcURI + ", notifyAction=" + notifyAction

LogUtils.d(TAG, "onRecvActionNotify srcURI=" + srcURI + ", notifyAction=" + notifyAction

+ ", notifyCode=" + notifyCode + ", notifyInfo=" + notifyInfo

+ ", notifyCode=" + notifyCode + ", notifyInfo=" + notifyInfo

+ ", notifyParams=" + notifyParams);

+ ", notifyParams=" + notifyParams);

if (notifyCode == ActionCode.ACTION\_OK) {

if (notifyCode == ActionCode.ACTION\_OK) {

//成功

//success

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

showToast("取消导航成功");

showToast ("cancel navigation successful");

}

}

});

});

} else {

} else {

//错误，错误码：notifyCode, 错误信息：notifyInfo

//Error, error code: notifyCode, error information: notifyInfo

mHandler.post(new Runnable() {

mHandler.post(new Runnable() {

@Override

@Override

public void run() {

public void run() {

showToast("取消导航失败 " + notifyCode + ", " + notifyInfo);

showToast("cancel navigation failed" + notifyCode + ", " + notifyInfo);

}

}

});

});

}

}

}

}

});

});

}

}

## 机器人控制API 11. Robot Control API

1. 初始化API实例

1) Initialize API instance

RobotActionCenterApi api = new RobotActionCenterApiImp();

1. 运动

2) Move

void moveForward();

void moveForward();

void turnLeft();

void turnLeft();

void turnRight();

void turnRight();

void moveBack();

void moveBack();

1. 停止移动

3) Stop move

void stopWalking();

void stopWalking();

void turnRound();

void turnRound();

1. 跳舞

4) Dance

void robotDance();

void robotDance();

void robotStopDance();

void robotStopDance();

1. 跟随

5) Follow

void robotFollow();

void robotFollow();

void stopFollow();

void stopFollow();

1. 停止全部，停止导航，唱歌跳舞等等

6) Stop all, stop navigation, sing and dance, etc.

void stopAll();

void stopAll();

1. 头部

Head

void turnHeadRight();

void turnHeadRight();

void turnHeadLeft();

void turnHeadLeft();

void turnHeadUp();

void turnHeadUp();

void turnHeadDown();

void turnHeadDown();

void turnHeadReset();

void turnHeadReset();

void headUpDown();

void headUpDown();

void headLeftRight();

void headLeftRight();

1. 灯环

8) Light ring

void lightNormal();

void lightNormal();

void lightTalking();

void lightTalking();

void lightThinking();

void lightThinking();

void lightListening();

void lightListening();

void lightSinging();

void lightSinging();

1. 启动默认任务

9) Start default task

void taskStartDefault(int startIndex, ActionNotifyCallBack callBack);

void taskStartDefault(int startIndex, ActionNotifyCallBack callBack);

1. 启动任务根据名称

10) Start task by name

void taskStartByName(String taskName, int startIndex, ActionNotifyCallBack callBack);

void taskStartByName(String taskName, int startIndex, ActionNotifyCallBack callBack);

1. 启动任务根据id

11) Start task by id

void taskStartById(String taskId, int startIndex, ActionNotifyCallBack callBack);

void taskStartById(String taskId, int startIndex, ActionNotifyCallBack callBack);

1. 停止任务

12) Stop task

void taskStop();

void taskStop();

1. 暂停任务

13) Pause task

void taskPause();

void taskPause();

## 导航控制API 12. Navigation Control API

RobotNavActionApiImp api = new RobotNavActionApiImp();

1. 开启导航

1) Open navigation

int startNav(ActionNotifyCallBack callBack);

int startNav(ActionNotifyCallBack callBack);

int startNavByMapId(String mapId, ActionNotifyCallBack callBack);

int startNavByMapId(String mapId, ActionNotifyCallBack callBack);

1. 停止导航

2) Stop navigation

int stopNav(ActionNotifyCallBack callBack);

int stopNav(ActionNotifyCallBack callBack);

1. 取消导航

3) Cancel navigation

int cancelNav(ActionNotifyCallBack callBack);

int cancelNav(ActionNotifyCallBack callBack);

1. 导航到某个点

4) Navigate to point

int navToPointByName(String pointName, ActionNotifyCallBack callBack);

int navToPointByName(String pointName, ActionNotifyCallBack callBack);

1. 根据坐标导航到某个点

5) Navigate to point by position

int navToPointByPostion(float x, float y, float z, ActionNotifyCallBack callBack);

int navToPointByPostion(float x, float y, float z, ActionNotifyCallBack callBack);

1. 获取导航状态

6) Get navigation state

int getNavState(ActionNotifyCallBack callBack);

int getNavState(ActionNotifyCallBack callBack);

1. 回冲

7) Docking

int backDock(ActionNotifyCallBack callBack);

int backDock(ActionNotifyCallBack callBack);

1. 取消回冲

8) Cancel docking

int cancelBackDock(ActionNotifyCallBack callBack);

int cancelBackDock(ActionNotifyCallBack callBack);

## 系统Action 13. System Action

系统Action，包括音频，视频，视频呼叫等等；

System Actions, including music, video, video call, etc.;

SystemActionCenterApiImp api = new SystemActionCenterApiImp();

SystemActionCenterApiImp api = new SystemActionCenterApiImp();

1. 播放音频

1) Play music

int playMusic(String url, ActionNotifyCallBack callBack);

int playMusic(String url, ActionNotifyCallBack callBack);

void resumeMusic();

void resumeMusic();

void pauseMusic();

void pauseMusic();

void stopMusic();

void stopMusic();

1. 播放视频

2) Play video

int playVideo(String videoUrl, String title, boolean isMute, ActionNotifyCallBack callBack);

int playVideo(String videoUrl, String title, boolean isMute, ActionNotifyCallBack callBack);

void resumeVideo();

void resumeVideo();

void pauseVideo();

void pauseVideo();

void stopVideo();

void stopVideo();

1. 图片轮播

3) Show pictures

int showPicsStart(List<String> imageList);

int showPicsStart(List<String> imageList);

void showPicsStop();

void showPicsStop();

1. 视频聊天

4) Video chat

/\*\*

/\*\*

\* 呼叫好友

\* Call friend

\* @param acountId

\* @param acountId

\* @param name

\* @param name

\* @param type

\* @param type

\*/

\*/

void callTo(String acountId, String name, int type);

void callTo(String acountId, String name, int type);

/\*\*

/\*\*

\* 呼叫好友

\* Call friend

\* @param params

\* @param params

\*/

\*/

void callTo(SystemCallParams params);

void callTo(SystemCallParams params);

/\*\*

/\*\*

\* 跳转到好友列表

\* Jump to friends list

\*/

\*/

void jumpFriendList();

void jumpFriendList();

## 打开/关闭唤醒的方法(只适用A1 A2 2.0版本) 14. Method of Open/Close Wakeup (version 2.0 only)

艾米桌面切换为对话桌面或者笑脸桌面时(艾米ASR开启)，麦克风被艾米语音桌面占用，第三方APP需要录音时，需要调用麦克风唤醒关闭的方法。

When the Amy desktop is switched to the dialog or smiley desktop (Amy ASR is opened), the microphone is occupied by the Amy voice desktop. When the third-party APP needs to record, it needs to call the method of the microphone to close wakeup.

打开:

Open:

Settings.System.*putInt*(getContext().getContentResolver(), **"is\_enble\_microphone"**,1);

Settings.System.*putInt*(getContext().getContentResolver(), **"is\_enble\_microphone"**,1);

关闭:

Close:

Settings.System.*putInt*(getContext().getContentResolver(), **"is\_enble\_microphone"**,0);

Settings.System.*putInt*(getContext().getContentResolver(), **"is\_enble\_microphone"**,0);

注意：A. 调用关闭方法后，最好延迟2s再调用录音功能，以便唤醒功能关闭完成

Notes: A. After calling the close method, it is best to delay 2s and then call the recording function, so as to close wake-up function.

艾米桌面切换为APP应用列表桌面时(艾米ASR关闭)，麦克风未被占用，第三方APP无需处理开关。

When the Amy Desktop switches to the APP App List desktop (Amy ASR is closed), the microphone is not occupied, and the third-party APP does not need to handle the switch.

## 监听唤醒broadcast 15. Monitor Wakeup Broadcast

对机器人讲：”hi amy ” ”hello amy”时，系统会发送com.amyrobotics.wakeup广播，第三方app监听此广播即可。

When saying "hi amy""hello amy” the robot, the system will send a com.amyrobotics.wakeup broadcast, and the third party app can monitor the broadcast.

## 第三方开发语音app时注意事项 16. Considerations When Developing a Voice App by a Third Party

如果只是将艾米语音桌面demo替换为开发者自己的demo，可参考Amina 2.0 user manual 3.1.5，替换为自己的ASR 和dialogflow账号.

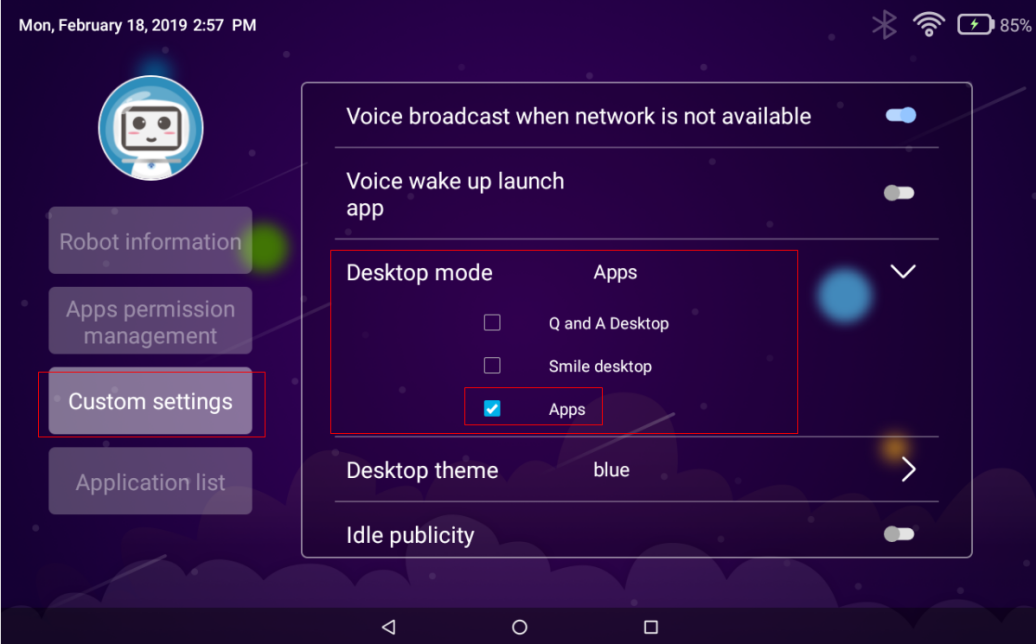
If you just want to replace the Amy Voice Desktop demo with the developer's own demo, refer to Amina 2.0 user manual 3.1.5, replace with your own ASR and dialogflow accounts.

第三方开发语音APP方式：

Third party development voice APP mode:

先艾米桌面应切换为APP应用列表桌面(艾米ASR关闭)。自己的APP中接入所选择的ASR平台处理语音识别，拿到文字后自行处理。APP需要自行监听唤醒broadcast，参考第15条。

Amy desktop should be switched to the APP list desktop first (Amy ASR closed). Call ASR service selected by user’s APP to process voice recognition, and then handle the text. APP needs to monitor the wake-up broadcast by itself, refer to Section 15.



## AmyRobot A1 OS v1.0(已停止维护)启用/禁用系统语音的方法 17. AmyRobot A1 OS v1.0 (maintenance stopped) Method to Enable/Disable System Voice

依次执行下方命令，即可禁用amy voice

Disable amy voice by executing the following commands sequentially.

adb root

adb root

adb remount

adb remount

adb shell

adb shell

cd /system/app/VoiceProvider

cd /system/app/VoiceProvider

mv VoiceProvider.apk VoiceProvider

mv VoiceProvider.apk VoiceProvider

cd /system/app/AmyVoiceZh

cd /system/app/AmyVoiceZh

mv AmyVoiceZh.apk AmyVoiceZh

mv AmyVoiceZh.apk AmyVoiceZh

reboot

reboot

依次执行下方命令，即可启用amy voice

Enable amy voice by executing the following commands sequentially.

adb root

adb root

adb remount

adb remount

adb shell

adb shell

cd /system/app/VoiceProvider

cd /system/app/VoiceProvider

mv VoiceProvider VoiceProvider.apk

mv VoiceProvider VoiceProvider.apk

cd /system/app/AmyVoiceZh

cd /system/app/AmyVoiceZh

mv AmyVoiceZh AmyVoiceZh.apk

mv AmyVoiceZh AmyVoiceZh.apk

reboot

reboot

## 电量广播及获得电量的方法 18. Power Broadcast & Method of Getting Power

远程控制中的“自动回充”设置打开时，低电时会强制中断任务，去执行充电任务。

When the "Auto Recharge" setting in the remote control is opened, task is force interrupted during low power to execute the charging task.

开发者如果不希望低电回冲打断自己APP的任务，可以先将远程控制中的“自动回充”设置关闭，开发APP时自行处理低电行为即可。

If the developer does not want the low-power recharge to interrupt the task of his/her own APP, he/she can close the "Auto Recharge" setting in the remote control first, and handle the low-power action when developing APP.

1. 充电完成时，会发送如下广播

When the charging is completed, the following broadcast will be sent,**private static final** String ***CHARGE\_COMPLETE\_CAST*** =

**private static final** String ***CHARGE\_COMPLETE\_CAST*** = **"com.amyrobotics.battery.chargeComplete"**;

**"com.amyrobotics.battery.chargeComplete"**;**private static**

**private static final**String***CHARGE\_COMPLETE\_CAST***=**"com.amyrobotics.battery.chargeComplete"**;

**final**String***CHARGE\_COMPLETE\_CAST***=**"com.amyrobotics.battery.chargeComplete"**;

1. 低电量时，会发送如下广播(当电量小于等于15%则判定为低电量)  
   When the power level is low, the following broadcast will be sent (the power level is determined to be low when it is less than or equal to 15%),  
   **private static final** String ***CHARGE\_LOW\_CAST*** =

**private static final** String ***CHARGE\_LOW\_CAST*** =

**"com.amyrobotics.battery.chargeLower"**;

**"com.amyrobotics.battery.chargeLower"**;

**private static**

**private static final**String***CHARGE\_LOW\_CAST***=**"com.amyrobotics.battery.chargeLower"**;

**final**String***CHARGE\_LOW\_CAST***=**"com.amyrobotics.battery.chargeLower"**;

1. 电量广播，包括电量值与充电状态

Power broadcast, including power value and charge status  
**private static final** String ***CHARGE\_CHANGE*** =

**private static final** String ***CHARGE\_CHANGE*** = **"com.amyrobotics.batteryChange"**;

**"com.amyrobotics.batteryChange"**;

**private static**

**private static final**String***CHARGE\_CHANGE***=**"com.amyrobotics.batteryChange"**;

**final**String***CHARGE\_CHANGE***=**"com.amyrobotics.batteryChange"**;

从intent中读出电量值与充电状态参考代码

Read power value and charge status reference code from the intent

**int** level = intent.getIntExtra(**"level"**, 0);

**int** level = intent.getIntExtra(**"level"**, 0);  
**int** charge = intent.getIntExtra(**"charging"**, 0);

**int** charge = intent.getIntExtra(**"charging"**, 0);

1. 获得电量及充电状态api如下

Get power level and charge status API as follows

**public static final** String ***APP\_SRC\_URI*** =

**public static final** String ***APP\_SRC\_URI*** = **"content://com.amyrobotics.amyservice.provider"**;

**"content://com.amyrobotics.amyservice.provider"**;

*/\*\**

*/\*\*  
 \* 获得电量Action，任何app调用此uri，可获得机器人当前电量。*

*\* Get Power Action, any app calls this uri to get the current power of the robot.  
 \*/*

*\*/***public static final** String ***ACTION\_NAME\_GET\_POWER\_LEVEL***=

**public static final** String ***ACTION\_NAME\_GET\_POWER\_LEVEL***= **"com.amyrobotics.amyservice.get.power.level"**;

**"com.amyrobotics.amyservice.get.power.level"**;  
*/\*\**

*/\*\*  
 \* 获得充电状态Action，任何app调用此uri，可获得机器人当前是否在充电。*

*\*Get Charge Status Action, any app calls this uri to get whether the robot is currently charging.  
 \*/*

*\*/***public static final** String ***ACTION\_NAME\_GET\_CHARGE\_STATE***=

**public static final** String ***ACTION\_NAME\_GET\_CHARGE\_STATE***= **"com.amyrobotics.amyservice.get.charge.state"**;

**"com.amyrobotics.amyservice.get.charge.state"**; *\*/***public static**

*\*/***public static final**String***ACTION\_NAME\_GET\_POWER\_LEVEL***=**"com.amyrobotics.amyservice.get.power.level"**;*/\*\**

**final**String***ACTION\_NAME\_GET\_POWER\_LEVEL***=**"com.amyrobotics.amyservice.get.power.level"**;*/\*\*   
 \*/*

*\*/***public static final** String ***ACTION\_NAME\_GET\_CHARGE\_STATE*** =

**public static final** String ***ACTION\_NAME\_GET\_CHARGE\_STATE*** = **"com.amyrobotics.amyservice.get.charge.state"**;

**"com.amyrobotics.amyservice.get.charge.state"**;

ActionResult actionResult =

ActionResult actionResult = ActionCenterManager.*getInstance*().executeAction(APP\_SRC\_URI,

ActionCenterManager.*getInstance*().executeAction(APP\_SRC\_URI,  
***ACTION\_NAME\_GET\_POWER\_LEVEL***, 0, **""**, **null**);

***ACTION\_NAME\_GET\_POWER\_LEVEL***, 0, **""**, **null**);  
LogUtils.*d*(***TAG***, **"current power level: "**+ actionResult.mVal);

LogUtils.*d*(***TAG***, **"current power level: "**+ actionResult.mVal);

ActionResult actionResult =

ActionResult actionResult = ActionCenterManager.*getInstance*().executeAction(APP\_SRC\_URI,

ActionCenterManager.*getInstance*().executeAction(APP\_SRC\_URI,  
***ACTION\_NAME\_GET\_CHARGE\_STATE***, 0, **""**, **null**);

***ACTION\_NAME\_GET\_CHARGE\_STATE***, 0, **""**, **null**);  
LogUtils.*d*(***TAG***, **"current charge state: "**+ actionResult.mVal);

LogUtils.*d*(***TAG***, **"current charge state: "**+ actionResult.mVal);

//1: charging

//1: charging