**艾米机器人远程控制SDK使用说明**

版本修改记录：

|  |  |  |  |
| --- | --- | --- | --- |
| 版本 | 修改说明 | 修改日期 | 修改人 |
| V1.0 | 新建SDK使用说明文档，包括导航，地图，任务，系统等命令； | 2019-02-19 | mudeyu  zhongjianhui |
| V1.1 | 修正命令名称和参数； | 2019-02-22 | mudeyu  zhongjianhui |
| V1.2 | 1. 添加建图接口；添加获取实时导航位置和实时建图数据接口；重连建图导航接口；  2. 添加获取系统状态，包括电量，是否开启导航，是否开启任务等等；  3. 添加充电，取消充电接口；  ​4. 添加导航代码示例，包括初始化，开始导航，获取位置，获取标记列表，导航到标记，停止导航；等等。。 | 2019-02-25 | mudeyu |
| V1.3 | 修正命令和参数； | 2019-03-07 | mudeyu |
| V1.4 | 修正命令和参数； | 2019-03-11 | mudeyu |
| V1.5 | 添加重置位置接口 | 2019-03-22 | mudeyu |

目录

[1. 概述 6](#_Toc4159391)

[2. 环境搭建 7](#_Toc4159392)

[2.1. 搭建开发环境 7](#_Toc4159395)

[2.2. 项目工程引入库 7](#_Toc4159398)

[2.3. 机器人端开启远程控制服务 7](#_Toc4159399)

[2.4. 通过局域网远程控制 8](#_Toc4159400)

[2.5. 通过云服务器远程控制 8](#_Toc4159401)

[3. 远程控制管理 9](#_Toc4159402)

[3.1. 创建远程控制管理对象 9](#_Toc4159404)

[3.2. 初始化参数 9](#_Toc4159405)

[3.3. 配置地图参数 9](#_Toc4159406)

[3.4. 监听连接事件 10](#_Toc4159407)

[3.5. 发送注册 11](#_Toc4159408)

[3.6. 日志设置 11](#_Toc4159409)

[3.7. 发送动作(指令) 12](#_Toc4159410)

[3.8. 监听机器人事件 13](#_Toc4159411)

[3.9. 服务开启与停止 14](#_Toc4159412)

[4. 建图 14](#_Toc4159413)

[4.1. 开始建图 14](#_Toc4159415)

[4.2. 停止建图并且不保存 15](#_Toc4159416)

[4.3. 停止建图并且保存建图 15](#_Toc4159417)

[4.4. 获取实时建图数据 15](#_Toc4159418)

[4.5. 获取建图状态 16](#_Toc4159419)

[4.6. 重连导航建图 16](#_Toc4159420)

[5. 导航 17](#_Toc4159421)

[5.1. 开启导航 17](#_Toc4159423)

[5.2. 停止导航 18](#_Toc4159424)

[5.3. 获取导航状态 19](#_Toc4159425)

[5.4. 导航状态 19](#_Toc4159426)

[5.5. 获取导航实时位置 19](#_Toc4159427)

[5.6. 导航到标记点 20](#_Toc4159428)

[5.7. 取消导航 21](#_Toc4159429)

[5.8. 设置默认导航地图 21](#_Toc4159430)

[5.9. 重置位置 22](#_Toc4159431)

[6. 地图管理 23](#_Toc4159432)

[6.1. 获取地图列表 23](#_Toc4159434)

[6.2. 删除地图 24](#_Toc4159435)

[6.3. 修改地图信息 24](#_Toc4159436)

[6.4. 获取地图数据 25](#_Toc4159437)

[6.5. 获取标记点列表 25](#_Toc4159438)

[6.6. 保存标记点列表 26](#_Toc4159439)

[7. 任务管理 26](#_Toc4159440)

[7.1. 获取任务列表 26](#_Toc4159442)

[7.2. 创建或修改保存任务 28](#_Toc4159443)

[7.3. 删除任务 30](#_Toc4159444)

[7.4. 开始任务&恢复任务 30](#_Toc4159445)

[7.5. 停止任务 31](#_Toc4159446)

[7.6. 暂停任务 31](#_Toc4159447)

[7.7. 执行下一步 32](#_Toc4159448)

[7.8. 执行上一步 32](#_Toc4159449)

[7.9. 设置默认任务 32](#_Toc4159450)

[7.10. 任务测试开关 33](#_Toc4159451)

[8. 机器人控制指令 33](#_Toc4159452)

[8.1. 前进 33](#_Toc4159454)

[8.2. 后退 34](#_Toc4159455)

[8.3. 向左转 34](#_Toc4159456)

[8.4. 向右转 34](#_Toc4159457)

[8.5. 停止移动 35](#_Toc4159458)

[8.6. 停止全部 35](#_Toc4159459)

[8.7. 跳舞 35](#_Toc4159460)

[8.8. 停止跳舞 36](#_Toc4159461)

[8.9. 机器人移动 36](#_Toc4159462)

[8.10. 头部转动 37](#_Toc4159463)

[8.11. 头部转右 37](#_Toc4159464)

[8.12. 头部转左 38](#_Toc4159465)

[8.13. 头部转上 38](#_Toc4159466)

[8.14. 头部转下 39](#_Toc4159467)

[8.15. 头部复位 39](#_Toc4159468)

[8.16. 点头 39](#_Toc4159469)

[8.17. 摇头 40](#_Toc4159470)

[8.18. 设置灯环 40](#_Toc4159471)

[8.19. 开始充电 41](#_Toc4159472)

[8.20. 取消充电 41](#_Toc4159473)

[9. 系统指令 41](#_Toc4159474)

[9.1. 获取媒体音量 41](#_Toc4159476)

[9.2. 设置媒体音量 42](#_Toc4159477)

[9.3. 音量加 42](#_Toc4159478)

[9.4. 音量减 43](#_Toc4159479)

[9.5. 说话&停止说话 43](#_Toc4159480)

[9.6. 唱歌 43](#_Toc4159481)

[9.7. 停止唱歌 44](#_Toc4159482)

[9.8. 音乐-播放音乐 44](#_Toc4159483)

[9.9. 音乐-暂停音乐 45](#_Toc4159484)

[9.10. 音乐-继续播放音乐 45](#_Toc4159485)

[9.11. 音乐-停止播放音乐 45](#_Toc4159486)

[9.12. 视频-播放视频 46](#_Toc4159487)

[9.13. 视频-暂停视频 46](#_Toc4159488)

[9.14. 视频-继续播放视频 47](#_Toc4159489)

[9.15. 视频-停止播放视频 47](#_Toc4159490)

[9.16. 显示表情 47](#_Toc4159491)

[9.17. 隐藏表情 48](#_Toc4159492)

[9.18. 轮播图片-开始轮播 48](#_Toc4159493)

[9.19. 轮播图片-关闭轮播 49](#_Toc4159494)

[9.20. 获取系统状态 49](#_Toc4159495)

[9.21. 获取版本信息 50](#_Toc4159496)

[10. 错误码说明 50](#_Toc4159497)

[11. 导航代码示例 51](#_Toc4159498)

[12.1. 初始化 51](#_Toc4159502)

[12.2. 开启服务 52](#_Toc4159503)

[12.3. 停止服务 53](#_Toc4159504)

[12.4. 开启导航 53](#_Toc4159505)

[12.5. 获取实时位置 54](#_Toc4159506)

[12.6. 获取标记点列表 55](#_Toc4159507)

[12.7. 导航到标记点 57](#_Toc4159508)

[12.8. 停止导航 58](#_Toc4159509)

## 概述

艾米机器人SDK为开发者提供了艾米机器人控制命令接口。开发者可以使用SDK提供的接口对机器人进行运动控制，导航控制，头部控制，播放音视频，播放语音，表情，定制任务等等。

## 环境搭建



### 搭建开发环境

获取安装JDK，需要JDK版本大于等于1.7版本；



### 项目工程引入库

项目工程添加引入库AmyRobotRemoteClientLib.jar；

引入依赖库；

<dependency>

<groupId>com.googlecode.protobuf-java-format</groupId>

<artifactId>protobuf-java-format</artifactId>

<version>1.2</version>

</dependency>

<dependency>

<groupId>com.google.protobuf</groupId>

<artifactId>protobuf-java</artifactId>

<version>3.0.0</version>

</dependency>

<dependency>

<groupId>com.alibaba</groupId>

<artifactId>fastjson</artifactId>

<version>1.2.18</version>

</dependency>

<dependency>

<groupId>io.netty</groupId>

<artifactId>netty-all</artifactId>

<version>4.1.27.Final</version>

</dependency>

### 机器人端开启远程控制服务

机器人端开启远程控制服务，并确认版本号大于等于 1.21.1；

### 通过局域网远程控制

连接成功，并且注册成功之后，通过sendAction方法发送命令；



### 通过云服务器远程控制

dstcid 设置为null，连接成功之后通过 sendRegister 发送注册，注册成功之后，通过sendAction方法发送命令；



## 远程控制管理



### 创建远程控制管理对象

RobotClientMgr mRobotClientMgr = RobotClientMgr.getInstance();

RobotClientMgr 提供方法来进行与服务器通信，与机器人通信；具体如下；

### 初始化参数

**方法名称：**setClientType(String clientType)

**方法描述：**设置client类型；类型不正确将会无法访问；

**方法参数：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数值** | **参数类型** | **参数说明** |
| CLIENT\_TYPE\_WEB | web | String | web端 |
| CLIENT\_TYPE\_MOBILE | mobile | String | 移动端 |

**代码示例：**

mRobotClientMgr.setClientType(RemoteClientType.CLIENT\_TYPE\_WEB);

**方法名称：**init (serverIP, serverPort, userName, cid, dstCid, passwd)

**方法描述：**初始化设置服务器IP地址，端口等；

**方法参数：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| serverIP | String | 是 | 服务器IP |
| serverPort | String | 是 | 服务器端口 |
| userName | String | 否 | 用户名称 |
| cid | String | 是 | 发送者id |
| dstCid | String | 是 | 机器人id |
| passwd | String | 是 | 密码 |

### 配置地图参数

**方法名称：**setUseMapDir (String mapRootDir);

**方法描述：**设置地图保存目录；

**方法参数：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| mapRootDir | String | 是 | 地图保存目录，可以是相对路径或者绝对路径； |

**代码示例：**

String mapRootDir= "amyrobot/map/";

mRobotClientMgr.setUseMapDir(mapRootDir);

**方法名称：**String getUseMapDir(String mapRootDir);

**方法描述：**获取地图保存目录；

**返回结果：**返回地图保存目录;

### 监听连接事件

**方法名称：**

addDataClientListener (String obj, DataClientListenerlistener);

removeDataClientListener(String obj);

**方法描述：**

addDataClientListener 添加连接事件监听；

removeDataClientListener 移除事件监听；

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| obj | String | 是 | String对象，用于标识不同的监听，可以设置监听多个； |
| listener | DataClientListener | 是 | 监听事件回调，见DataClientListener |

**DataClientListener接口说明：**

|  |  |
| --- | --- |
| **接口名称** | **接口说明** |
| onConnected | 连接成功回调; |
| onRegisterResult | 注册结果回调;  srcCId 注册cid；  code 错误码；200表示成功，其他表示出错；  errInfo 如果出错，返回出错信息； |
| onDisconnected | 连接已经断开; |
| onError | 出错回调; |

**代码示例：**

String DATA\_HANDLER\_CALLBACK\_ID = "10000";

mRobotClientMgr.addDataClientListener(DATA\_HANDLER\_CALLBACK\_ID, new BaseDataClientListener() {

@Override

public void onConnected() {

LogUtils.d(TAG, "onConnected");

}

@Override

public void onRegisterResult(String srcCId, int code, String errInfo) {

LogUtils.d(TAG, "onRegisterResult: " + srcCId + ", " + code + ", " + errInfo);

handleRegisterResult(srcCId, code, errInfo);

}

@Override

public void onDisconnected() {

LogUtils.d(TAG, "onDisconnected");

}

@Override

public void onError(Throwable e) {

LogUtils.e(TAG, "onError", e);

}

});

### 发送注册

**方法名称：**void sendRegister(String cid, String pwd);

**方法描述：**连接成功之后，通过发送注册接口给机器人，只有注册成功才能进行控制，否则不能进行控制;

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| cid | String | 是 | 机器人id |
| pwd | String | 是 | 密码 |

**返回结果：**

返回结果会在 RobotEventListener 接口回调 onRegisterResult();

### 日志设置

**方法名称：**setDebug (boolean enable)

**方法描述：**设置启用或禁用调试日志打印；日志输出到console；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| enable | boolean | 是 | 是否启用调试日志 |

**方法名称：**setLogLevel (int level)

**方法描述：**设置日志打印级别；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| level | int | 是 | LOG\_LEVEL\_VERBOSE 全部  LOG\_LEVEL\_DEBUG 调试  LOG\_LEVEL\_INFO 信息  LOG\_LEVEL\_WARN 警告  LOG\_LEVEL\_ERROR 错误 |

### 发送动作(指令)

**方法名称：**

boolean sendAction(String action, Map<String, String> params, ActionEventCallback requestCallback);

boolean sendAction(String dstId, String action, Map<String, String> params, ActionEventCallback requestCallback);

boolean sendAction(String srcId, String dstId, String action, Map<String, String> params, ActionEventCallback requestCallback);

**命令描述：**发送动作指令；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| srcId | String | 是 | 发送者id |
| dstCid | String | 是 | 接收者id |
| action | String | 是 | 动作指令名，具体根据不同动作对应不同名称； |
| params | Map<String, String> | 否 | 根据具体action 传递不同参数，不需要参数直接传null； |
| requestCallback | ActionEventCallback | 否 | 成功回调: onSuccess(RobotEvent robotEvent);  具体action 还需判断resultCode错误码，参考错误码说明；  robotEvent.resultCode == CODE\_OK  失败回调：onFailed(int code, String msg, Throwable e); |

**返回结果：**根据不同action返回不同结果；

**代码示例：**

String srcId = "200100";

String dstCid = "200101";

// startNavigation

String action = "robot.startNavigation";

Map<String, String> params = new HashMap<>();

params.put("mapId", "069a6c1d-e0fb-4d5b-a8fa-5b4c9a0b2cf5");

mRobotClientMgr.sendAction(srcId, dstCid, action, params, new ActionEventCallback() {

@Override

public void onSuccess(RobotEvent robotEvent) {

if(robotEvent.resultCode == RobotNotifyCode.CODE\_OK) {

LogUtils.e(TAG, "ok " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

} else {

LogUtils.e(TAG, "error " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

}

}

@Override

public void onFailed(int code, String msg, Throwable e) {

LogUtils.e(TAG, "error " + code + ", " + msg + ", " + e);

}

});

### 监听机器人事件

**方法名称：**setRobotEventListener(RobotEventListener listener)

**方法描述：**设置机器人事件监听；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| listener | RobotEventListener | 是 | 机器人事件回调；  void onRobotEvent(RobotEvent event);  RobotEvent参数见下； |

RobotEvent 属性说明:

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| cid | String | 是 | 机器人id |
| notifyAction | String | 是 | 动作名称 |
| resultCode | int | 是 | 返回结果错误码，200表示成功，其他表示失败，具体参考错误码说明； |
| notifyInfo | String | 否 | 错误信息，不同Action错误信息不同； |
| notifyParams | String | 否 | 返回结果 |
| notifyPktId | String | 是 | 事件通知id |
| data | Object | 否 | 额外数据，扩展用； |

**代码示例：**

mRobotClientMgr.setRobotEventListener(new RobotEventListener() {

@Override

public void onRobotEvent(RobotEvent robotEvent) {

handleRobotEvent(robotEvent);

}

});

### 服务开启与停止

**方法名称：**start();

**方法描述：**启动服务；根据配置服务器地址自动连接服务器，在连接事件监听接收连接事件；

**方法名称：**stop();

**方法描述：**停止服务；断开与服务器连接，停止监听事件；

**方法名称：**boolean isConnected ();

**方法描述：**获取是否已经与服务器连接；

## 建图



### 开始建图

**命令名称：**robot.startCreateMap

**命令描述：**开始建图；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| mapName | String | 是 | 地图名称，地图名称不能重复 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.startCreateMap |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明；  如果地图名称已存在返回错误码 CODE\_ERR\_NAME\_DUPLICATE |
| notifyInfo | String | 错误信息 |
| notifyParams | String | ok：开启成功 |

### 停止建图并且不保存

**命令名称：**robot.stopCreateMap

**命令描述：**停止建图并且不保存地图；

**参数描述：无**

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.stopCreateMap |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 无 |

### 停止建图并且保存建图

**命令名称：**robot.saveMapAndStopMapMode

**命令描述：**停止建图并保存地图；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| name | String | 是 | 地图名称 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.saveMapAndStopMapMode |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明；  如果地图名称已存在返回错误码 CODE\_ERR\_NAME\_DUPLICATE |
| notifyInfo | String | 错误信息 |
| notifyParams | String | mapId |

### 获取实时建图数据

开始建图之后，将通过 RobotEvent 发送给Client实时建图数据；

RobotEvent 具体信息:

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.notifyMsg.mapData |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 地图数据json |

实时地图数据：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| topic | String | “/map\_posting” 表示是地图数据， |
| msg | json | 见下 |
| op | String | pubish 表示发布 |

msg 参数如下：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| map | String | 地图数据 |
| width | int | 地图宽度 |
| height | int | 地图高度 |
| x | double | x坐标 |
| y | double | y坐标 |
| size | double | size |

### 获取建图状态

**命令名称：**robot.getCreateMapStatus

**命令描述：**获取建图状态；

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.getCreateMapStatus |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 是否已经开启建图。  0: 表示停止建图；  1: 表示开始建图； |

### 重连导航建图

**命令名称：**robot.reconnectNav

**命令描述：**重连导航建图，在开启建图或开启导航之后，网络断开，再次连接注册成功之后，发送重连导航建图命令来重新接收实时地图和实时位置信息；

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.reconnectNav |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

## 导航



### 开启导航

**命令名称：**robot.startNavigation

**命令描述：**开启导航，只有开启导航之后才能执行导航任务；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| mapId | String | 是 | 地图ID |
| useDefault | String | 否 | 是否使用默认地图；  如果为1，当mapId为空时，自动使用默认地图开启导航；  如果为0，必须设置mapId才能开启导航； |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.startNavigation |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明；  常见错误：  CODE\_ERROR\_STOP\_SWITCH\_IS\_OPEN, 开启导航失败，急停开关已经打开；  CODE\_ERROR, useMap failed, 使用地图失败，地图ID对应的地图文件不存在，地图文件数据或格式错误等等；具体看 notifyInfo； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | ok：开启导航成功  开始失败 |

**代码示例：**

String srcId = "200100";

String dstCid = "200101";

String action = "robot.startNavigation";

Map<String, String> params = new HashMap<>();

params.put("mapId", "069a6c1d-e0fb-4d5b-a8fa-5b4c9a0b2cf5");

mRobotClientMgr.sendAction(srcId, dstCid, action, params, new ActionEventCallback() {

@Override

public void onSuccess(RobotEvent robotEvent) {

if(robotEvent.resultCode == RobotNotifyCode.CODE\_OK) {

LogUtils.e(TAG, "ok " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

} else {

LogUtils.e(TAG, "error " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

}

}

@Override

public void onFailed(int code, String msg, Throwable e) {

LogUtils.e(TAG, "error " + code + ", " + msg + ", " + e);

}

});

**注意事项：**

需要先打开急停开关才可以开启导航，机器人只能在起始位置开启导航否则无法正确导航；

### 停止导航

**命令名称：**robot.stopNavigation

**命令描述：**停止导航

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.stopNavigation |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

**注意事项：**

停止导航之后再开启导航，需要重新把机器人放置到起始位置；

### 获取导航状态

**命令名称：**robot.getNavState

**命令描述：**获取导航状态

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.getNavState |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 参见导航状态 |

导航状态：

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| **名称** | | **值** | | **类型** | | **说明** |
| NAVI\_START | #NAV02 | | String | | 导航开启 | |
| NAVI\_STOP | #NAV07 | | String | | 导航停止 | |

### 导航状态

导航状态定义：

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数值** | **参数类型** | **参数说明** |
| NAVI\_ARRIVE | #NAV01 | String | 导航到达 |
| NAVI\_START | #NAV02 | String | 导航开启 |
| NAVI\_LOST | #NAV03 | String | 导航丢失 |
| NAVI\_GIVEUP | #NAV04 | String | 导航放弃 |
| NAVI\_TIMEOUT | #NAV05 | String | 导航超时 |
| NAVI\_CANCLE\_SUCCESS | #NAV06 | String | 导航取消成功 |
| NAVI\_STOP | #NAV07 | String | 导航停止 |

### 获取导航实时位置

开启导航之后，将通过 RobotEvent 发送给Client实时位置信息

RobotEvent 具体信息:

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.notifyMsg.mapData |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 数据见下 |

实时位置数据：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| map | String | 地图数据 |
| topic | String | 为 “/robot\_hpose” 时表示是位置坐标数据： |
| msg | json | 实时位置坐标 |
| op | String | pubish 表示发布 |

实时位置坐标

|  |  |  |
| --- | --- | --- |
| 参数名称 | 参数类型 | 参数说明 |
| x | double | x坐标 |
| y | double | y坐标 |
| z | double | 方向，弧度 |

示例：

{

"topic": "/robot\_hpose",

"msg": {

"y": 0.0313592925667763,

"x": 1.09918606281281,

"z": -0.0359665118157864

},

"op": "publish"

}

### 导航到标记点

**命令名称：**robot.navToPoint

**命令描述：**导航到标记位置

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| x | float | 是 | x坐标 |
| y | float | 是 | y坐标 |
| z | float | 是 | 方向弧度 |
| name | String | 是 | 导航点名称，用于日志记录 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.navToPoint |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 导航到达时返回 NAVI\_ARRIVE=”#NAV01”; 返回其他值参见导航状态表； |

### 取消导航

**命令名称：**robot.cancelGoal

**命令描述：**取消导航去标记位置

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.cancelGoal |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 设置默认导航地图

**命令名称：**robot.setDefaultNavMap

**命令描述：**设置默认导航地图 ID，如果开启导航地图id为空，useDefault参数为1，并且已经设置默认地图，将使用默认地图来开启导航；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| mapId | String | 是 | 地图ID |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.setDefaultNavMap |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 无 |

### 重置位置

**命令名称：**robot.navResetPos

**命令描述：**根据坐标值，重置机器人当前位置；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| x | float | 是 | x坐标 |
| y | float | 是 | y坐标 |
| z | float | 是 | 方位弧度 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | cid |
| notifyAction | String | robot.navResetPos |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 无 |

**代码示例：**

public void resetPos(MarkPointEntity markPointEntity) {

Map<String, String> params = new HashMap<>();

params.put("x", String.valueOf(markPointEntity.realX));

params.put("y", String.valueOf(markPointEntity.realY));

float rad = (float) Math.toRadians(markPointEntity.realAngle);

params.put("z", String.valueOf(rad));

RobotClientMgr.getInstance().sendAction(RobotAction.NAV\_RESET\_POS, params, new ActionEventCallback() {

@Override

public void onSuccess(RobotEvent robotEvent) {

if(robotEvent != null) {

if(robotEvent.resultCode == RobotNotifyCode.CODE\_OK) {

LogUtils.d(TAG, "navResetPos ok");

} else {

LogUtils.d(TAG, "navResetPos failed " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

}

}

}

@Override

public void onFailed(int errCode, String errInfo, String result) {

LogUtils.d(TAG, "navResetPos error " + errCode + ", " + errInfo);

}

});

}

## 地图管理



### 获取地图列表

**命令名称：**robot.getMapList

**命令描述：**获取地图列表

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.getMapList |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 地图信息列表，json字符串，List<MarkPointEntity> |

MapListEntity：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| mapFileInfoList | List<MapFileInfo> | 地图信息列表 |
| defaultNavMapId | String | 默认地图id，如果为空表示没有设置默认地图； |

地图信息MapFileInfo：

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| id | String | 地图的id |
| name | String | 名称 |
| desc | String | 描述 |
| createTime | String | 创建时间 |
| width | Int | 地图宽度 |
| height | Int | 地图高度 |
| ratio | Double | 地图分辨率，1像素对应实际多少米 |
| x | Double | 初始位置的X |
| y | Double | 初始位置的Y |
| z | Double | 初始位置的Z |

### 删除地图

**命令名称：**robot.deleteMap

**命令描述：**删除地图

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| mapId | String | 是 | 地图ID |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.deleteMap |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 无 |

### 修改地图信息

**命令名称：**robot.modifyMap

**命令描述：**修改地图信息；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| mapId | String | 是 | 地图ID |
| newMapName | String | 是 | 修改后的名称 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.modifyMap |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 获取地图数据

**命令名称：**robot.getMapData

**命令描述：**获取地图数据；发送获取地图数据命令之后，会自动保存地图到设置的地图目录； setUseMapDir()设置地图保存目录；例如：

String mapRootDir= "amyrobot/map/";

mRobotClientMgr.setUseMapDir(mapRootDir);

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| mapId | String | 是 | 地图ID |
| mapLastModifiedTime | long | 否 | 手机端上次缓存地图修改时间，long类型，如果没有缓存地图，可以传0；  如果mapLastModifiedTime与当前地图修改时间相同，则不会重传地图文件； |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.getMapData.Complete接收地图完成； |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | mapId 地图id |

### 获取标记点列表

**命令名称：**robot.getMarkPointList

**命令描述：**获取标记点列表

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.getMarkPointList |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | json, List<MarkPointEntity> |

标记点实体MarkPointEntity：

|  |  |  |
| --- | --- | --- |
| **名称** | **参数类型** | **参数说明** |
| text | String | 标记名称，添加时判断是否重复，避免重复名称，用于导航 |
| desc | String | 标记描述信息 |
| name2 | String | 标记点别名 |
| x | float | 地图坐标x |
| y | float | 地图坐标Y |
| realX | float | 实际坐标x |
| realY | float | 实际坐标Y |
| realAngle | float | 实际角度，使用 Math.toRadians(realAngle) 转换成弧度； |
| isStartpoint | boolean | 是否是起始点，起始点不能删除； |
| angle | float | 图上角度 |
| radian | float | 图上弧度 |

### 保存标记点列表

**命令名称：**robot.saveMarkPoint

**命令描述：**保存标记列表

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| mapId | String | 是 | 地图ID |
| markPointList | String | 是 | json，List<MarkPointEntity> 具体参考 MarkPointEntity |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.saveMarkPoint |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

## 任务管理



### 获取任务列表

**命令名称：**robot.getTaskList

**命令描述：**获取任务列表

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.roamTaskList |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 任务列表实体参考RoamTaskListEntity |

任务列表实体对象RoamTaskListEntity：

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| taskEntityList | List<RoamTaskEntity> | 任务实体对象列表 |
| isTaskRun | boolean | 是否有任务在运行 |
| runTaskId | String | 正在运行的任务id |
| defaultTaskId | String | 默认漫游任务id |

任务实体对象 RoamTaskEntity：

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| taskId | String | 任务id |
| name | String | 漫游名称 |
| mapId | String | 漫游对应的地图id |
| mapName | String | 地图名称 |
| roamPointEntityList | List<RoamPointEntity> | 漫游点列表 |
| cycleTime | Int | 循环次数 -1表示无限循环 |
| arriveDo | boolean | 是否必须到达之后再进行后续动作（语音播报） |
| params | Map<String, String> | 添加扩展参数 |

漫游任务步骤点实体RoamPointEntity

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| pointName | String | 漫游点名称 |
| action | String | 漫游到达执行动作，如果为null 或者 空，默认表示语音播报;  参考 TaskActions |
| strVal | String | String参数 |
| Params | Map<String, String> | 扩展参数 |
| paramList | Int | List参数 |
| actionPreDelayMs | Int | Action执行之前延时，单位毫秒 |
| actionDelayMs | boolean | Action执行之后延时，单位毫秒 |
| childAction | List<TaskActionEntity> | 子action, 当主action执行时，会同时启动子action，主action结束时，同时停止子action |

子Action TaskActionEntity：

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| action | String | Action名称 |
| strVal | String | Action参数 |
| params | Map<String, String> | 扩展参数，根据不同action参数不同 |
| paramList | List<String> | list参数，根据不同action参数不同 |

TaskActions

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数值** | **参数类型** | **参数说明** |
| TASK\_ACTION\_NONE | none | String | 无动作 |
| TASK\_ACTION\_VOICE | voice | String | 语音播报 |
| TASK\_ACTION\_AUDIO | audio | String | 播放音频 |
| TASK\_ACTION\_VIDEO | video | String | 播放视频 |
| TASK\_ACTION\_SPEECH\_RECOGNITION | speech.recognition | String | 语音识别 |
| TASK\_ACTION\_BACK\_CHARGE | robot.backCharge | String | 充电 |
| TASK\_ACTION\_SHOW\_PICS | showPics | String | 播放视频 |
| TASK\_ACTION\_TASK\_CONFIRM | taskConfirm | String | 任务确认 |
| TASK\_ACTION\_EXT\_CMD | ext.cmd | String | 扩展指令 |
| TASK\_ACTION\_START\_TASK | robot.startTask | String | 停止当前任务，启动新任务 |

### 创建或修改保存任务

**命令名称：**robot.taskSave

**命令描述：**保存任务

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| mode | String | 是 | create: 创建;  modify: 修改; |
| taskEntity | String | 是 | 任务实体对象，RoamTaskEntity |

RoamTaskEntity 类：

public class RoamTaskEntity implements Serializable {

public static final int TASK\_CYCLE\_INFINITE = -1; //无限循环

public String taskId; //任务id

public String name; //漫游名称

public String mapId; //漫游对应的地图 Id

public String mapName; //地图名称

public List<RoamPointEntity> roamPointEntityList; //漫游点列表

public int cycleTimes; //循环次数, -1 表示无限循环

public boolean arriveDo; //是否必须到达之后再进行后续动作（语音播报），true 表示必须到达之后再执行，false 表示即使未到达也会执行后续动作

public Map<String, String> params; //添加扩展参数

}

public class RoamPointEntity implements Serializable {

public String pointName; //漫游点名称

public String action; //漫游到达执行动作，如果为null 或者 空，默认表示语音播报

public String strVal; //string 参数

public Map<String, String> params; //扩展参数

public List<String> paramList; //list 参数

public int actionPreDelayMs; //action 执行之前延时，单位毫秒

public int actionDelayMs; //action 执行之后延时，单位毫秒

public List<TaskActionEntity> childActions; //子action, 当主action执行时，会同时启动子action，主action结束时，同时停止子action

}

public class TaskActionEntity implements Serializable {

public String action; //action 名称

public String strVal; //action 参数

public Map<String, String> params; //扩展参数

public List<String> paramList; //list 参数

}

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.taskSave |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 删除任务

**命令名称：**robot.taskDelete

**命令描述：**删除

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| taskId | String | 是 | 任务Id |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.taskDelete |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 开始任务&恢复任务

**命令名称：**robot.taskStart

**命令描述：**开始任务，如果任务处于暂停中会恢复任务执行；如果任务已经开始，会直接返回，开始新任务需要先停止任务再开始任务；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| taskId | String | 是 | 任务Id |
| useDefault | String | 否 | 是否使用默认任务；  1 如果任务id为空，会开始默认任务；  0 表示不会开始默认任务； |
| index | int | 是 | 从任务第几步开始执行，默认为0; |
| isJumpStep | Int | 是 | 是否跳转执行；  如果isJumpStep=0，如果任务暂停，表示是恢复任务执行，不管index为多少；  如果isJumpStep=1,会先停止当前执行步骤，再执行index对应的步骤； |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **参数名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.taskStart |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 停止任务

**命令名称：**robot.taskStop

**命令描述：**停止任务

**参数描述：无**

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.taskStop |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 暂停任务

**命令名称：**robot.taskPause

**命令描述：**暂停任务

**参数描述：无**

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.taskPause |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 执行下一步

**命令名称：**robot.taskNextStep

**命令描述：**执行下一步

**参数描述：无**

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.taskNextStep |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 执行上一步

**命令名称：**robot.taskPrevStep

**命令描述：**执行上一步

**参数描述：无**

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.taskPrevStep |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 设置默认任务

**命令名称：**robot.setDefaultTask

**命令描述：**设置默认任务

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| taskId | String | 是 | 任务Id |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.setDefaultTask |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 任务测试开关

**命令名称：**robot.taskTest

**命令描述：**设置任务测试开关；任务测试开关打开，开启任务不会自动开启导航，不会执行导航到标记点，只会执行动作；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| taskTest | Int | 是 | 1表示打开，0表示关闭； |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.taskTest |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |

## 机器人控制指令



### 前进

**命令名称：**robot.goForward

**命令描述：**前进

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.goForward |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |

### 后退

**命令名称：**robot.moveBack

**命令描述：**后退

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.moveBack |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 向左转

**命令名称：**robot.turnLeft

**命令描述：**向左转

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.turnLeft |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 向右转

**命令名称：**robot.turnRight

**命令描述：**向右转

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.turnRight |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 停止移动

**命令名称：**robot.stopWalking

**命令描述：**停止移动

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.stopWalking |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 停止全部

**命令名称：**robot.stopAll

**命令描述：**停止全部（停止移动，停止导航，停止唱歌，停止跳舞，停止漫游）

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.stopAll |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 跳舞

**命令名称：**robot.dance

**命令描述：**跳舞

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.dance |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 停止跳舞

**命令名称：**robot.stopDance

**命令描述：**停止跳舞

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.stopDance |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 机器人移动

**命令名称：**robot.move

**命令描述：机器人**移动

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| linearVelocity | float | 是 | 线速度 |
| angularVelocity | float | 是 | 角速度 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.move |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

代码示例：

void sendMove(float linearVelocity, float angularVelocity) {

Map<String, String> params = new HashMap<>();

params.put("linearVelocity", String.valueOf(linearVelocity));

params.put("angularVelocity", String.valueOf(angularVelocity));

sendAction(RobotAction.ROBOT\_MOVE, params);

}

### 头部转动

**命令名称：**robot.turnHead

**命令描述：**转动头部，当 upDown和 leftRight 为0时，表示停止头部转动；

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| upDown | int | 是 | 上负下正。取值范围 -255~255 |
| leftRight | int | 是 | 左负右正。取值范围 -255~255 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.turnHead |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

代码示例：

void turnHead(int upDown, int leftRight) {

Map<String, String> params = new HashMap<>();

params.put("upDown", String.valueOf(upDown));

params.put("leftRight", String.valueOf(leftRight));

sendAction(RobotAction.TURN\_HEAD, params);

}

### 头部转右

**命令名称：**robot.turnHeadRight

**命令描述：**头部向右转

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.turnHeadRight |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 头部转左

**命令名称：**robot.turnHeadLeft

**命令描述：**头部向左转

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.turnHeadLeft |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 头部转上

**命令名称：**robot.turnHeadUp

**命令描述：**头部向上转

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.turnHeadUp |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 头部转下

**命令名称：**robot.turnHeadDown

**命令描述：**头部向下转

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.turnHeadDown |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 头部复位

**命令名称：**robot.turnHeadReset

**命令描述：**头部复位

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.turnHeadReset |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 点头

**命令名称：**robot.headUpDown

**命令描述：**点头

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.headUpDown |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 摇头

**命令名称：**robot.headLeftRight

**命令描述：**摇头

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.headLeftRight |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 设置灯环

**命令名称：**robot.light

**命令描述：**设置灯环

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| type | String | 是 | 灯环类型  lightNormal 正常  lightTalking 说话  lightThinking 思考  lightSinging 唱歌 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.light |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 开始充电

**命令名称：**robot.backDock

**命令描述：**开始充电，将机器人对准充电桩，发送该指令会让机器人开始充电；

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.backDock |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 取消充电

**命令名称：**robot.cancelBackDock

**命令描述：**取消充电

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.cancelBackDock |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

## 系统指令



### 获取媒体音量

**命令名称：**sys.getMusicVolume

**命令描述：**获取媒体音量

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.getMusicVolume |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 设置媒体音量

**命令名称：**sys.setMusicVolume

**命令描述：**设置媒体音量

**参数描述：**

|  |  |  |  |
| --- | --- | --- | --- |
| **参数名称** | **参数类型** | **是否必须** | **参数说明** |
| volume | Int | 是 | 音量 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.setMusicVolume |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 音量加

**命令名称：**sys.setVolumeAdd

**命令描述：**音量加

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.setVolumeAdd |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 音量减

**命令名称：**sys. setVolumeDec

**命令描述：**音量减

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.setVolumeDec |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 说话&停止说话

**命令名称：**sys.say

**命令描述：**发送说话文本；停止说话，直接发送标点比如逗号，就会停止；

**参数描述：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| words | String | 语音转音频的文字 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.say |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 唱歌

**命令名称：**robot.singsong

**命令描述：**唱歌

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.singsong |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 停止唱歌

**命令名称：**robot.stopSing

**命令描述：**停止唱歌

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.stopSing |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 音乐-播放音乐

**命令名称：**sys.player.music.play

**命令描述：**播放音乐

**参数描述：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| url | String | 音频的url  例如：<http://xx/1.mp3>  /sdcard/music/1.mp3 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.player.music.play |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 音乐-暂停音乐

**命令名称：**sys.player.music. pause

**命令描述：**暂停音乐

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.player.music. pause |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 音乐-继续播放音乐

**命令名称：**sys.player.music. resume

**命令描述：**继续播放音乐

**参数描述：无**

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.player.music. resume |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 音乐-停止播放音乐

**命令名称：**sys.player.music. stop

**命令描述：**停止播放音乐

**参数描述：**

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.player.music. stop |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 视频-播放视频

**命令名称：**sys.player.video.play

**命令描述：播放视频**

**参数描述：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| url | String | 视频的url  例如：<http://xx/1.mp4>  /sdcard/video/1.mp4 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.player.video.play |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 视频-暂停视频

**命令名称：**sys.player.video.pause

**命令描述：暂停视频**

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.player.video. pause |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 视频-继续播放视频

**命令名称：**sys.player.video.resume

**命令描述：继续播放视频**

**参数描述：无**

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.player.video.resume |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 视频-停止播放视频

**命令名称：**sys.player.video.stop

**命令描述：停止播放视频**

**参数描述：无**

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.player.video. stop |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 显示表情

**命令名称：**sys.showFace

**命令描述：**显示表情

**参数描述：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| faceType | String | 表情类型；  faceTalk: 说话表情  faceSmail: 笑脸表情 |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.showFace |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 隐藏表情

**命令名称：**sys.hideFace

**命令描述：**隐藏表情

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys. hideFace |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 轮播图片-开始轮播

**命令名称：**sys.showpics.start

**命令描述：**开始轮播图片

**参数描述：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| picList | List<String> | 图片的url的list;  例如：http://xx/1.png |

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.showpics.start |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 轮播图片-关闭轮播

**命令名称：**sys.showpics.stop

**命令描述：**关闭轮播

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | sys.showpics.stop |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String |  |

### 获取系统状态

**命令名称：**robot.getSystemState

**命令描述**：获取系统信息状态

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.getSystemState |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 系统状态 |

系统状态实体

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| chargePercent | float | 电量百分比 (0-100) |
| isCharging | boolean | 是否在充电 |
| isTaskRun | boolean | 是否任务运行 |
| isStartCreateMap | boolean | 是否开启建图 |
| isStartNav | boolean | 是否开启导航 |
| isDance | boolean | 是否在跳舞 |

### 获取版本信息

**命令名称：**robot.getVersion

**命令描述**：获取版本信息

**参数描述：**无

**返回结果：**

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| cid | String | 机器人id |
| notifyAction | String | robot.getVersion |
| resultCode | int | 错误码，200表示成功，其他表示出错具体参考错误码说明； |
| notifyInfo | String | 错误信息 |
| notifyParams | String | 版本信息； |

版本信息；

|  |  |  |
| --- | --- | --- |
| **返回名称** | **参数类型** | **参数说明** |
| versionCode | int | 远程控制版本号 |
| versionName | String | 远程控制版本名称 |
| buildTime | String | 远程控制打包时间 |
| IPCVersion | String | 工控版本 |

## 错误码说明

|  |  |  |
| --- | --- | --- |
| **参数名称** | **错误码** | **说明** |
| CODE\_OK | 200 | 执行成功 |
| CODE\_ERROR\_INVALID\_PARAMS | 300 | 参数错误 |
| CODE\_ERROR | 400 | 普通错误，具体要看notifyInfo 说明 |
| CODE\_ERROR\_TIMEOUT | 401 | 超时错误 |
| CODE\_ERROR\_IO | 402 | 文件读写错误 |
| CODE\_ERR\_NAME\_EMPTY | 403 | 名称不能为空 |
| CODE\_ERR\_NAME\_DUPLICATE | 405 | 名称重复 |
| CODE\_ERROR\_INVALID\_PARAMS | 406 | 无效参数 |
| CODE\_ERROR\_STOP\_SWITCH\_IS\_OPEN | 407 | 急停开关已经打开 |
| CODE\_ERROR\_INVALID\_CMD | 408 | 无效命令 |
| CODE\_ERROR\_RESULT\_EMPTY | 409 | 结果为空错误 |
| CODE\_ERROR\_ALREADY\_START | 410 | 已经开始，必须先停止才能开始 |

## 导航代码示例



### 初始化

public static final String DATA\_HANDLER\_CALLBACK\_ID = "1000";

public static final String ROBOT\_IP = "192.168.0.100";

public static final int ROBOT\_PORT = 8889;

RobotClientMgr mRobotClientMgr;

String mRobotIP = ROBOT\_IP;

int mRobotPort = ROBOT\_PORT;

String mUserName = "amy";

String cid = "20001001"; //my cid

String dstCid = "10003011"; //robot cid

String passwd = null;

public void init() {

//设置log打印

LogUtils.setDebug(true);

//设置打印所有

LogUtils.setLogLevel(LogUtils.LOG\_LEVEL\_VERBOSE);

mRobotClientMgr = RobotClientMgr.getInstance();

mRobotClientMgr.addDataClientListener(DATA\_HANDLER\_CALLBACK\_ID, new BaseDataClientListener() {

@Override

public void onConnected() {

LogUtils.d(TAG, "onConnected");

}

@Override

public void onRegisterResult(String srcCId, int code, String errInfo) {

handleRegisterResult(srcCId, code, errInfo);

}

@Override

public void onDisconnected() {

LogUtils.d(TAG, "onDisconnected");

}

@Override

public void onError(Throwable e) {

LogUtils.e(TAG, "onError", e);

}

});

cid = UUID.randomUUID().toString();

mRobotClientMgr.setClientType(RemoteClientType.CLIENT\_TYPE\_WEB);

mRobotClientMgr.init(mRobotIP, mRobotPort, mUserName, cid, dstCid, passwd);

LogUtils.d(TAG, "cid =" + cid + ", dstCid=" + dstCid);

LogUtils.d(TAG, "mRobotIP =" + mRobotIP + ", " + mRobotPort);

String mapRootDir = "amy/robot/";

mRobotClientMgr.setUseMapDir(mapRootDir);

}

### 开启服务

public void start() {

if (mRobotClientMgr.isRun()) {

LogUtils.d(TAG, "isRun = true, must stop");

return;

}

mRobotClientMgr.setRobotEventListener(new RobotEventListener() {

@Override

public void onRobotEvent(RobotEvent robotEvent) {

handleRobotEvent(robotEvent);

}

});

mRobotClientMgr.start();

LogUtils.d(TAG, "start ok");

}

### 停止服务

public void stop() {

mRobotClientMgr.stop();

while (mRobotClientMgr.isRun()) {

try {

Thread.sleep(1000);

} catch (InterruptedException e) {

e.printStackTrace();

}

}

LogUtils.d(TAG, "stop ok");

}

### 开启导航

String srcId = "200100";

String dstCid = "200101";

String mapId = "069a6c1d-e0fb-4d5b-a8fa-5b4c9a0b2cf5";

public void startNav() {

String action = RobotAction.START\_NAVIGATION;

Map<String, String> params = new HashMap<>();

params.put("mapId", mapId);

mRobotClientMgr.sendAction(cid, dstCid, action, params, new ActionEventCallback() {

@Override

public void onSuccess(RobotEvent robotEvent) {

if(robotEvent.resultCode == RobotNotifyCode.CODE\_OK) {

LogUtils.d(TAG, "START\_NAVIGATION ok");

} else {

if (RobotNotifyCode.CODE\_ERROR\_STOP\_SWITCH\_IS\_OPEN == robotEvent.resultCode) {

LogUtils.d(TAG, "START\_NAVIGATION error CODE\_ERROR\_STOP\_SWITCH\_IS\_OPEN ");

} else {

LogUtils.d(TAG, "START\_NAVIGATION error " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

}

}

}

@Override

public void onFailed(int code, String msg, Throwable e) {

LogUtils.d(TAG, "error " + code + ", " + msg + ", " + e);

}

});

}

### 获取实时位置

// setRobotEventListener

mRobotClientMgr.setRobotEventListener(new RobotEventListener() {

@Override

public void onRobotEvent(RobotEvent robotEvent) {

handleRobotEvent(robotEvent);

}

});

private void handleRobotEvent(RobotEvent robotEvent) {

if (robotEvent.notifyAction == null) {

return;

}

LogUtils.d(TAG, "onRobotEvent cid=" + robotEvent.cid

+ ", act=" + robotEvent.notifyAction

+ ", resultCode=" + robotEvent.resultCode

+ ", notifyInfo=" + robotEvent.notifyInfo

+ ", notifyParams=" + robotEvent.notifyParams

);

switch (robotEvent.notifyAction) {

case RobotAction.NOTIFY\_MSG\_MAP\_DATA: {

if (RobotNotifyCode.CODE\_SUCCESS == robotEvent.resultCode) {

try {

String data = robotEvent.notifyParams;

JSONObject jsonObject = JSONObject.parseObject(data);

String topic = jsonObject.getString("topic");

switch (topic) {

case RobotNavConstant.MAP\_POSE:

JSONObject robotPositon = jsonObject.getJSONObject("msg");

RobotPositionInfo rpi = new RobotPositionInfo();

rpi.x = robotPositon.getDouble("x");

rpi.y = robotPositon.getDouble("y");

rpi.z = robotPositon.getDouble("z");

LogUtils.d(TAG, "onRobotEvent cid=" + robotEvent.cid

+ ", x=" + rpi.x

+ ", y=" + rpi.y

+ ", z=" + rpi.z

+ ", degree=" + Math.toDegrees(rpi.z)

);

break;

}

} catch (Exception e) {

LogUtils.d(TAG, "error " + e);

}

}

}

break;

default:

break;

}

}

### 获取标记点列表

String mapId = "069a6c1d-e0fb-4d5b-a8fa-5b4c9a0b2cf5";

public void getMarkPointList(String mapId) {

String action = RobotAction.GET\_MARK\_POINT\_LIST;

Map<String, String> params = new HashMap<>();

params.put("mapId", StringUtils.safe(mapId));

mRobotClientMgr.sendAction(cid, dstCid, action, params, new ActionEventCallback() {

@Override

public void onSuccess(RobotEvent robotEvent) {

if(robotEvent.resultCode == RobotNotifyCode.CODE\_OK) {

LogUtils.d(TAG, "ok " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

String data = robotEvent.notifyParams;

if (!StringUtils.isEmpty(data)) {

List<MarkPointEntity> markPointList = null;

try {

markPointList = JSON.parseArray(data, MarkPointEntity.class);

} catch (Exception e) {

e.printStackTrace();

}

for (MarkPointEntity markPointEntity : markPointList) {

LogUtils.d(TAG, "onRobotEvent name=" + markPointEntity.text

+ ", desc=" + markPointEntity.desc

+ ", isStartPoint=" + markPointEntity.isStartPoint

+ ", realX=" + markPointEntity.realX

+ ", realY=" + markPointEntity.realY

+ ", realAngle=" + markPointEntity.realAngle

);

}

}

} else {

LogUtils.d(TAG, "error " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

}

}

@Override

public void onFailed(int code, String msg, Throwable e) {

LogUtils.d(TAG, "error " + code + ", " + msg + ", " + e);

}

});

}

### 导航到标记点

public void navToPoint(MarkPointEntity markPointEntity) {

Map<String, String> params = new HashMap<>();

params.put("x", String.valueOf(markPointEntity.realX));

params.put("y", String.valueOf(markPointEntity.realY));

params.put("z", String.valueOf(Math.toRadians(markPointEntity.realAngle)));

mRobotClientMgr.sendAction(cid, dstCid, RobotAction.NAV\_TO\_POINT, params, new ActionEventCallback() {

@Override

public void onSuccess(RobotEvent robotEvent) {

LogUtils.d(TAG, "onRobotEvent cid=" + robotEvent.cid

+ ", act=" + robotEvent.notifyAction

+ ", resultCode=" + robotEvent.resultCode

+ ", notifyInfo=" + robotEvent.notifyInfo

+ ", notifyParams=" + robotEvent.notifyParams

);

if(robotEvent.resultCode == RobotNotifyCode.CODE\_OK) {

LogUtils.d(TAG, "ok IPCResponse.NAVI\_ARRIVE ");

} else {

LogUtils.d(TAG, "error " + robotEvent.resultCode + ", " + robotEvent.notifyInfo + ", " + robotEvent.notifyParams);

String data = robotEvent.notifyParams;

switch (data) {

case IPCResponse.NAVI\_LOST:

break;

case IPCResponse.NAVI\_GIVEUP:

break;

case IPCResponse.NAVI\_TIMEOUT:

break;

case IPCResponse.NAVI\_CANCLE\_SUCCESS:

break;

case IPCResponse.NAVI\_STOP:

break;

}

}

}

@Override

public void onFailed(int code, String msg, Throwable e) {

LogUtils.d(TAG, "error " + code + ", " + msg + ", " + e);

}

});

}

### 停止导航

public void stopNav() {

String action = RobotAction.STOP\_NAVIGATION;

mRobotClientMgr.sendAction(cid, dstCid, action, null, new ActionEventCallback() {

@Override

public void onSuccess(RobotEvent robotEvent) {

if(robotEvent.resultCode == RobotNotifyCode.CODE\_OK) {

LogUtils.e(TAG, "ok " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

} else {

LogUtils.e(TAG, "error " + robotEvent.resultCode + ", " + robotEvent.notifyInfo);

}

}

@Override

public void onFailed(int code, String msg, Throwable e) {

LogUtils.e(TAG, "error " + code + ", " + msg + ", " + e);

}

});

}