CreaAPI接口文档说明

1 文档说明

creabot.py 是基于Websocket接口开发的、针对Python编程语言的API。提供使用Python获取机器人状态信息、设计控制机器人表现的能力，您可以轻松使用python代码控制机器人。

通用底盘机器人(CreaBot)开源控制协议

2 准备工作

编程环境

准备一台配有无线网卡的电脑，并安装好python开发环境。

IDE推荐 VSCode（官网：https://code.visualstudio.com/） 、PyCharm（官网：https://www.jetbrains.com/pycharm/）、 Jupyter（官网：https://jupyter.org/）

建议在开发前在开发用的电脑上配置python开发虚拟环境，使用anaconda或virtualenv。

编程语言

Python 3.8.0及以上版本

软件依赖

将creabot-0.2-py3-none-any.whl文件拷贝到对应工程目录下/Python开发软件依赖库的路径下

使用pip安装依赖库:

```
pip config set global.index-url https://pypi.tuna.tsinghua.edu.cn/simple
```

```
pip install websocket-client
```

```
pip install requests
```

```
pip install --force-reinstall creabot-0.2-py3-none-any.whl
```

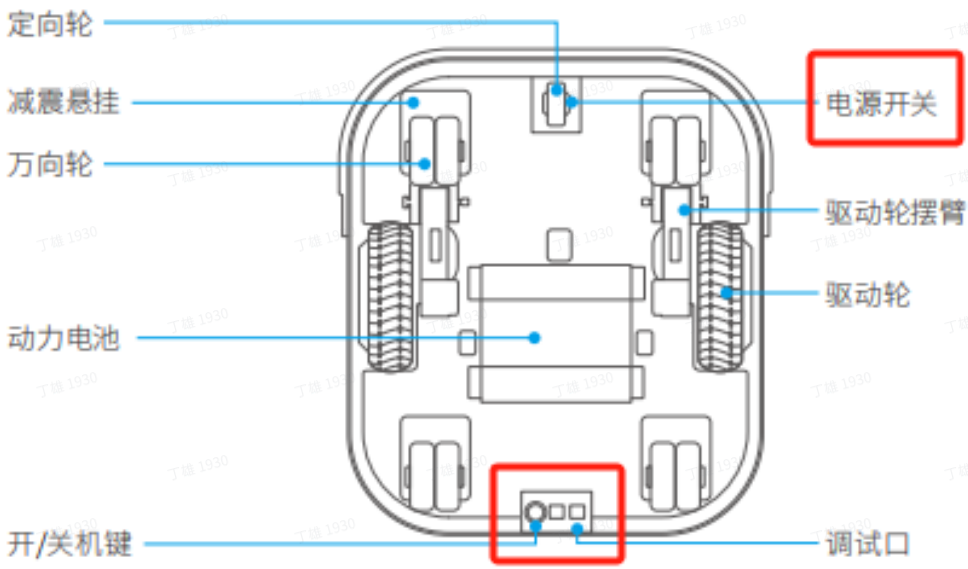
```
pip install base64
```

```
pip install time
```

开机

电源开关位于机器人正面底部，打开电源后，机器人会进入待机状态。

开机按钮位于机器人背面底部，长按5s开机，启动机器人。



网络配置

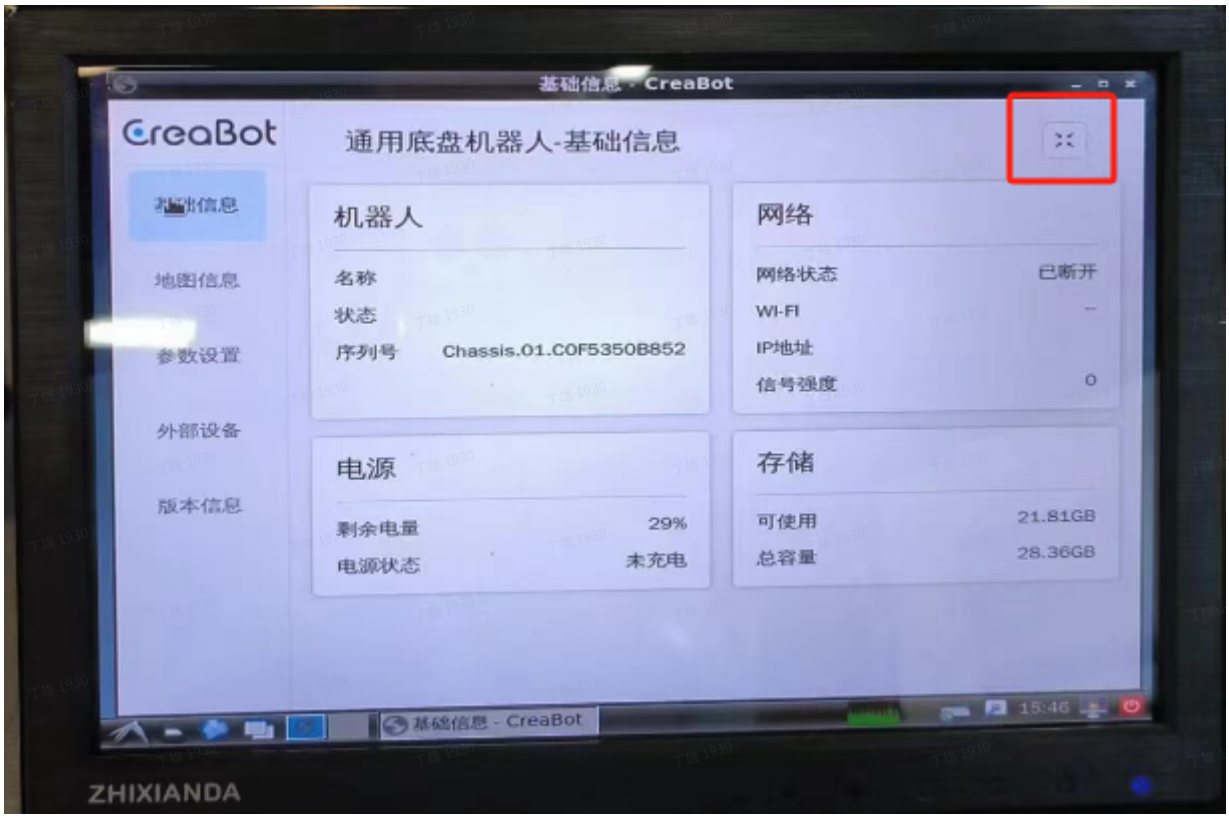
机器人支持wifi，第一次开机使用机器人，需要配置机器人网络。配置流程：

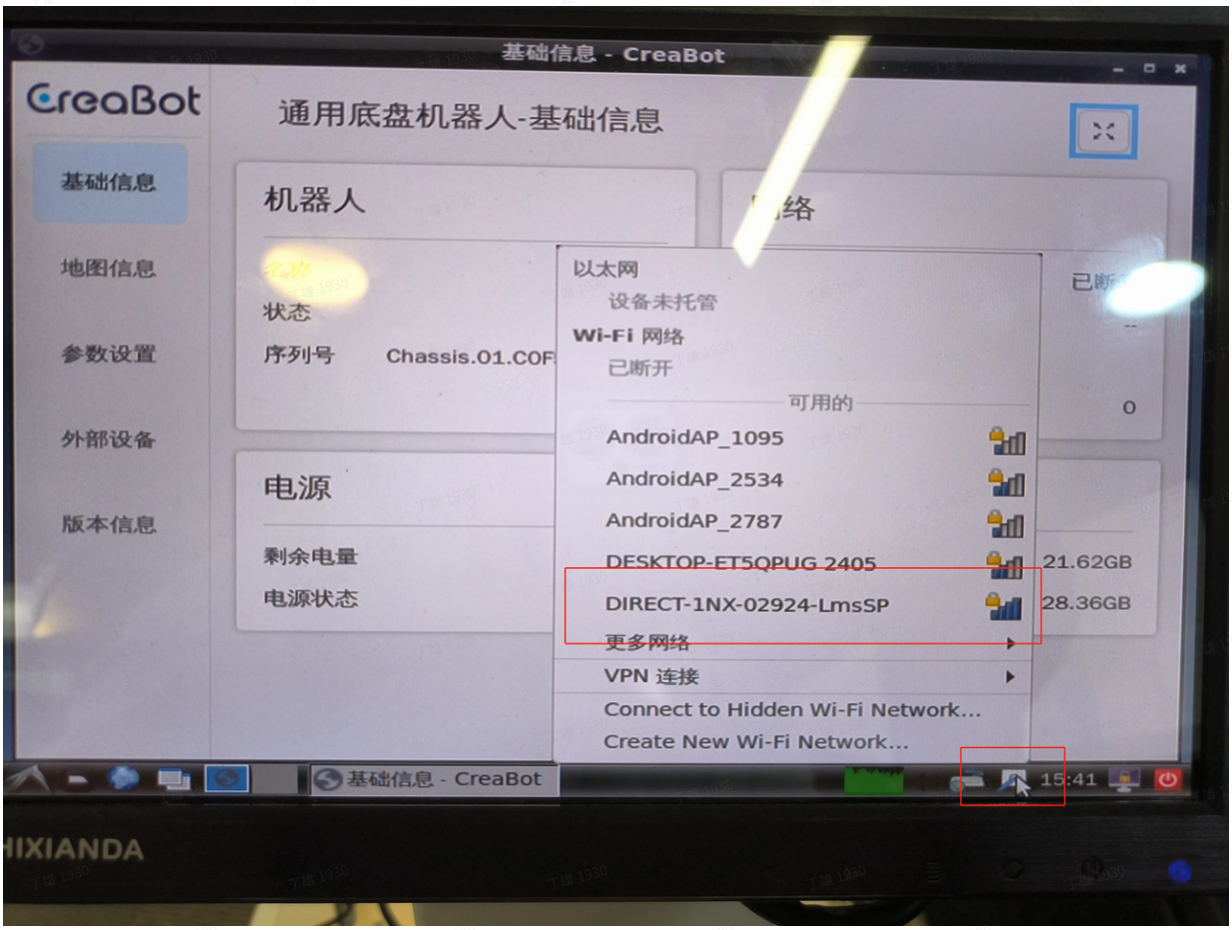
在机器人背部拔掉原有的USB连接线，然后接入鼠标键盘；

鼠标点击机器人控制台右上角“退出全屏”按钮，然后最小化控制台，退出到机器人系统桌面；

鼠标单击屏幕右下角网络图标，为机器人配置wifi网络，将机器人连入到和电脑同一个Wi-Fi，并等待连接网络成功；

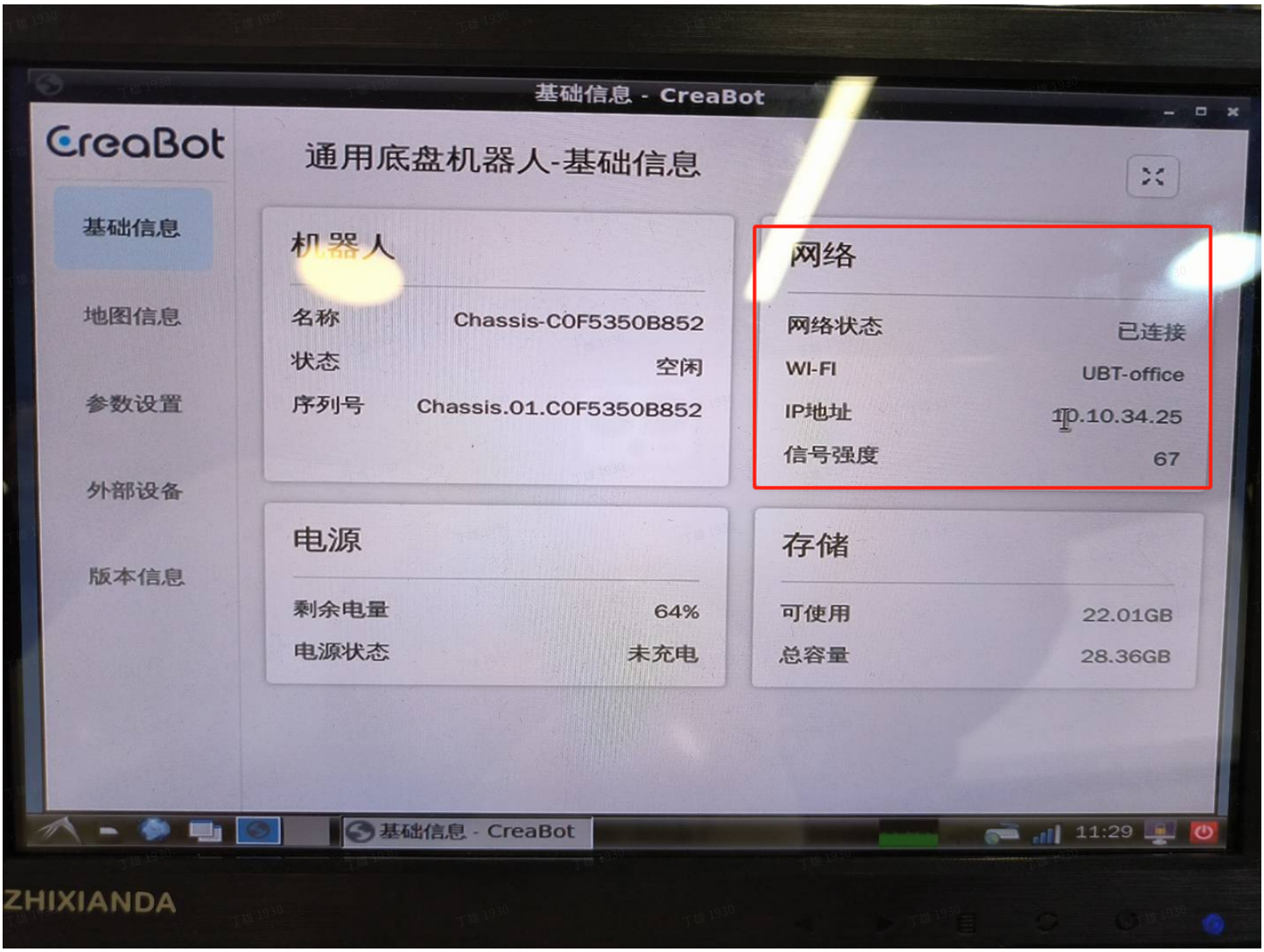
完成wifi设置后，拔出鼠标键盘，接入原有的连接线，然后回到机器人控制台，点击“版本信息”的重启按钮，重启机器人。





获取IP地址

查看机器人控制台，在基础信息查看“网络”，获得机器人在当前局域网的IP地址



示例代码

如果输入以下代码，获得类似的返回数据，则说明已经成功与机器人通信，可以开始基于python进行开发。

```python
ip_addr = "192.168.1.1" # 替换为自己的IP地址
bot=creabot.Creabot(ip_addr)
maplist = bot.list_map()
print(maplist)
```

{'cmd': 'response_map_list', 'code': 0, 'data': {'maps': [{'id': '03ff3292-46e6-4b6d-ac18-e94e3
bd0f', 'name': '展馆地图0820-02'}, {'id': '830dc742-ec83-4b27-b18d-5840abce15e9', 'name': '展馆
图0820-01'}, {'id': '7b5a43ec-120e-49b0-8e20-7eb35ac1de11', 'name': '展会地图0814-PM-01'}, {'id
'0b97d84e-fb87-4161-8384-4da916ba4a14', 'name': '0814-01'}, {'id': 'dda43db6-47e8-47fa-ae0f-5e2
139fa7', 'name': 'T888'}, {'id': '8bc917e2-df69-49ad-b60e-8f6c18baf742', 'name': 'T7'}, {'id':
da2260-d5ea-4e78-be14-c2211c310562', 'name': '0812赛事'}, {'id': '7043a7eb-cf7b-4ec7-90e5-0d312
4e07', 'name': '123'}, {'id': '36f7b389-e761-464e-81f5-31dda6a59aaa', 'name': '0812-01'}, {'id'
ace4aaf5-c4bd-4be3-af12-aeed4e2c970f', 'name': 'test8.9'}, {'id': '07c0bcda-7948-4092-af9c-f61e
0c553', 'name': 'test8.9'}, {'id': '0a372acb-8228-402b-a480-31547a9d5966', 'name': '0802-1434.0
, {'id': '36823a90-ddaa-4f1a-b3b3-601027f36383', 'name': '0807-sdk调试地图'}, {'id': '783cc52b-
f-4a37-ab9b-b3f51f690328', 'name': '0805赛事2'}, {'id': '6624bb75-8036-41d2-885d-639fe6907b06',
ame': '0805赛事'}, {'id': '17aeff13-ea41-484e-a2f2-63a4df443a89', 'name': '导航测试'}, {'id': '
5f7a7-cebb-4aad-9a24-48fac836e17e', 'name': '0802赛事2'}, {'id': '70ccb76b-f05d-48d0-aeb4-296e2
930c', 'name': '0802赛事'}, {'id': '76d42b6a-da15-48c4-9173-20636742f15d', 'name': '建图测试'},
id': '29a2440d-3e29-4d27-855f-af726a1a706d', 'name': '老化0801'}], 'total': 20}, 'msg': 'succes

3 状态码

以下为通用底盘机器人常用的状态码介绍

4 接口说明

4.1 导航

4.1.1 查询地图列表

list_map()

接口描述：查询机器人本体保存的地图列表

参数说明

无

返回类型

dict

返回说明

{

"cmd": "response_map_list", 命令标识符

"code": 0, 响应状态码， `0` 表示成功

"msg": "success", 响应消息，成功时为 `"success"`

"data": {

"maps": [{

"id": 地图的唯一标识符（UUID）

"name": 地图在云平台保存的名称

}]

"total": 地图的数量

}

}

4.1.2 设置默认地图

set_map(map_id)

接口描述：设置指定地图作为机器人的导航地图，只有设置地图的机器人才能使用导航相关功能和接口。设置地图后，机器人需要进行重定位。

参数说明

map_id: list_map()接口中获取的地图的唯一标识符（UUID）

返回类型

dict

返回说明：

{

"cmd": "response_set_map", 命令标识符

"code": 0, 响应状态码， `0` 表示成功，其他

"msg": "success", 响应消息，成功时为 `"success"` ，error

"data": {}

}

4.1.3 查询位置点列表

list_map_point(map_id)

接口描述：查询指定地图的位置点列表，位置点包含每一个点位的坐标和方向信息。

参数说明

map_id: list_map()接口中获取的地图的唯一标识符（UUID）

返回类型

dict

返回说明：

{

"cmd": str "response_point_list", 命令标识符

"code": int 0, 返回状态码，0表示成功

"msg": str "success", 返回消息，通常为 "success"

"data": {

"mapId": 地图的唯一标识符（UUID）

"points": [

{

"id": ,位置点的唯一标识符（UUID），

"name": 云平台编辑地图时保存的点的名称

"type": "destination", 云平台编辑的点的类型（例如“destination”）

"x": 点在地图上的X坐标位置

"y": 150, 点在地图上的Y坐标位置

"theta": 0, 点的方向角度

}

],

"total": 点列表的总数

}

}

位置点类型说明

名称	代号	描述	备注
目的地	*destination*	目标点位，比如：送餐餐桌点、巡检的检测点	机器人导航的目的地点位，通常可以设置如：餐桌、递送目的地等。设计机器人导航线路时，建议选择目的地作为导航任务目的点。
充电桩	*charge*	充电桩点位	用于机器人自动上桩充电。注意确保充电桩前方1米范围内无障碍物遮挡，否则无法上桩充电
出餐点	*dining*	出餐点一般表示厨房出餐点，送餐模式下，结束任务默认返回出餐点。	送餐任务专用，如果设置了此点位，送餐任务开始前将先导航到这个点位。一般可将厨房设置为出餐点。送餐任务结束后，机器人默认返回该点位，如果该点位未设置，默认返回定位点。

回收点	*recycle*	机器人回收点，非送餐模式下，结束任务默认返回点	非送餐任务使用，导航任务结束后，机器人默认返回该点位，如果该点位未设置，默认返回定位点。
定位点	*anchor_point*	机器人在当前地图的定位点，每张地图必须设置一个地位点	必须设置一个定位点，切换地图或者编辑地图，需把机器人推到该点位进行重定位，重定位成功后才能开始导航任务。设置定位点后，注意确保定位点周边环境不要发生大的变动，否则会出现重定位失败。

4.1.4 导航到指定点位（同步控制指令）

start_navigation_sync(x,y,theta,speed)

接口描述：设置地图并重定位成功后，可以控制机器人导航到指定的点位，机器人的导航算法会规划最优路径，并自动避障，最终达到该点位。

参数说明：

x: 导航目标的X坐标

y: 导航目标的Y坐标

theta: 导航目标的方向

speed: 导航的速度（单位是m/s），速度范围：0.3--1.0 m/s

返回类型：

dict

返回说明：

{

"cmd": "response_start_navigation", 命令标识符，固定为 "response_start_navigation"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": {} 可选的返回数据，本例中为空对象

}

导航状态码：

导航过程中可以通过状态码获取机器人的实时状态，通常用来进行事件触发或异常处理。

状态码	枚举名称	描述
6000	NAVI_SUCCESS	导航成功
6001	NAVI_CANCEL	取消导航
6002	NAVI_INIT_FAILED	初始化失败
6003	NAVI_GET_POSE_FAILED	获取位置失败
6004	NAVI_GOAL_OUT_COSTMAP	目标点在地图外
6005	NAVI_GOAL_ON_STATIC_OBSTACLE	目标点在静态障碍物上
6006	NAVI_NO_PATH_STATIC	找不到可通行路径
6007	NAVI_GOAL_ON_DYNAMIC_OBSTACLE	目标点在动态障碍物上
6008	NAVI_NO_PATH_DYNAMIC	找不到可通行路径
6009	NAVI_START_ON_OBSTACLE	起点有障碍物
6010	NAVI_POSE_LOST	定位丢失
6011	NAVI_LOCAL_PLANNER_FAILED	局部路径规划失败
6012	NAVI_CLIFF	悬崖触发取消
6013	NAVI_EMERGENCY	急停触发取消
6014	NAVI_STRIP	防撞条触发取消
6015	NAVI_TIMEOUT	超时
6016	NAVI_NOPILEDETECTED	充电桩检测失败
6017	TO_PILE_FAILED	上桩失败
6018	NAVI_GLOBAL_PLAN_FAILED	全局路径规划失败
6019	NAVI_SLOPE	斜坡触发取消
6020	NAVI_GIVE_WAY	导航中，请注意避让
6021	NAVI_LIFT_TIME_OUT	进出电梯超时
6022	NAVI_LIFT_LONG_TIME_NARROW	进出电梯太窄
6023	NAVI_LIFT_GLOBAL_PATH_UNSTABLE	进出电梯路径不稳定
6024	NAVI_RECOVERY_FAILED	导航恢复行为失败

6025	NAVI_NARROW_CHANNEL_OUT	导航退出窄通道
6026	NAVI_SAFE_GOAL_OUT_TOLERANCE	可达安全点超出超出设定允许范围
6100	NAVI_RUNNING	导航运行中
6101	NAVI_NO_PATH	导航运行中没有路径
6102	NAVI_PAUSE	导航暂停
6999	NAVI_UNKNOWN_ERROR	其他error

4.1.5 导航到指定点位（异步控制指令）

start_navigation(x,y,theta,speed)

接口描述：同上 4.1.4

参数说明：同上 4.1.4

返回类型：同上 4.1.4

返回说明：同上 4.1.4

状态码：同上 4.1.4

4.1.6 取消导航

stop_navigation()

接口描述：取消导航任务，机器人停留在原地等待。

参数说明

无

返回类型

dict

返回说明：

{

"cmd": "response_stop_navigation", 命令标识符，固定为 "response_stop_navigation"

"code": 0, 返回状态码，0表示成功

"msg": "success", 返回消息，通常为 "success"

"data": {} 可选的返回数据，本例中为空对象

}

4.1.7 重定位（同步控制指令）

relocate_sync(x,y,theta)

接口描述：机器人设置机器人后，必须重新进行当前地图定位，以便机器人确定在地图中的位置。只有定位成功的机器人才能开启导航。

参数说明：

x: 导航目标的X坐标

y: 导航目标的Y坐标

theta: 导航目标的方向

返回类型

dict

返回说明：

{

"cmd": "response_relocate_position",命令标识符，固定为 "response_relocate_position"

"code": 0, 返回状态码，0表示成功

"msg": "success", 返回消息，通常为 "success"

"data": {} 可选的返回数据，本例中为空对象

}

重定位状态码：

状态码	枚举值	描述
4000	RELOCATE_SUCCESS	重定位成功
4001	RELOCATE_FAILED	重定位失败
4002	RELOCATE_CANCEL	重定位取消

4.1.8 重定位（异步控制指令）

relocate(x,y,theta)

接口描述：同上 4.1.7

参数说明：同上 4.1.7

返回类型：同上 4.1.7

返回说明：同上 4.1.7

状态码：同上 4.1.7

4.1.9 停止重定位

stop_relocate()

接口描述：停止机器人重定位

参数说明：

无

返回类型

dict

返回说明：

{

"cmd": "response_relocate_position",命令标识符，固定为 "response_relocate_position"

"code": 0, 返回状态码，0表示成功

"msg": "success", 返回消息，通常为 "success"

"data": {} 可选的返回数据，本例中为空对象

}

4.1.10 监听函数

once(event_name, handler)

接口描述：监听机器人任务执行时的状态

参数说明：

event_name：事件标志（状态码）

handler：触发函数，监听到事件标志执行该函数

4.2 外设控制

4.2.1 舱门控制

door_ctrl(option)

接口描述：底盘安装递送套件后，通过当前指令控制开关舱门。

注意：第一次安装使用送餐套件，需要在关门状态下，点击机器人控制台--外部设备--舱门--校准按钮，进行舱门位置校准。

参数说明：

option，控制选项，0表示关闭舱门，1 表示打开舱门

返回类型

dict

返回说明

{

"cmd": "response_door_ctrl", 命令标识符，固定为 "response_door_ctrl"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": {

"status": 1，返回的数据对象，包含舱门的状态，舱门的状态，0表示关闭，1表示打开

}

}

4.2.2 舱门状态

is_door_open()

接口描述：机器人安装送餐套件时，获取仓门当前的状态，是否开闭。

请求参数

无

应答说明

物体检测状态，False表示门关闭，True表示门打开

4.2.3 照明灯控制

light_ctrl(option)

接口描述：机器人安装送餐套件时，通过当前指令控制舱内照明灯亮灭。

参数说明：

option，控制选项，0表示关灯，1表示开灯

返回类型

dict

返回说明

{

"cmd": "response_light_ctrl", 命令标识符，固定为 "response_light_ctrl"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": {

"status": 1，返回的数据对象，灯光的状态，0表示关灯状态，1表示开灯状态

}

}

4.2.4 消杀控制

uv_ctrl(level)

接口描述：机器人安装消杀套件时，通过当前指令控制消杀灯亮灭。

参数说明：

level，控制选项，0表示关闭紫外线灯，1、2、3分别表示不同的消杀强度

返回类型

dict

返回说明：

{

"cmd": "response_uvlight_ctrl", 命令标识符，固定为 "response_uvlight_ctrl"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": {

"status": 2，返回的数据对象，包含紫外线灯的状态，0表示关闭，1、2、3分别表示不同的消杀强度

}

}

4.2.5 舱内物体检测

exist_object()

接口描述：机器人安装送餐套件时，机器人上报舱内红外的物体检测结果。

参数说明：

无

返回类型

bool

返回说明：

物体检测状态，False表示未检测到物体，True表示检测到物体

4.2.6 视频流控制

get_camera_stream(status)

接口描述：控制机器人视频流传输。打开视频流，机器人会间隔200ms返回一帧视频帧

参数说明：

status，控制选项，0表示关闭视频流，1表示开启视频流

返回类型：

dict

返回说明：

{

"cmd": "response_camera_stream",命令标识符，固定为 "notify_camera_stream"

"code": 0,返回状态码，0表示成功

"msg": "success",返回消息，通常为 "success"

"data": {

"stream": 视频流帧数组，每个元素代表一帧图像数据

"base64_encoded_image_data"

]

} 包含视频流的数据对象

}

4.2.7 摄像头拍照

take_photo()

接口描述：控制机器人拍摄一张照片，照片将转为base64编码数据。

参数说明：

无

返回类型：

dict

返回说明：

{

"cmd": "response_take_photo",命令标识符，固定为 "response_take_photo"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": { 包含照片数据的对象

"img": "iVBORw0KGgoAAAANSUhEUgAAAOEAAADh..." 拍摄的照片数据，通常为 base64 编码格式

}

}

4.2.8 获取RGBD照片

get_rgbd_image()

接口描述：控制机器人RGBD拍摄一张照片，机器人将返回深度图和RGB图的base64编码数据。

参数说明：

无

返回类型：

dict

返回说明：

{

"cmd": "response_rgbd_image",命令标识符，固定为 "response_rgbd_image"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": { 包含 RGB 和深度图像数据的对象

"depth": "iVBORw0KGgoAAAANSUhEUgAAAOEAAADh...", 深度图像数据，通常为 base64 编码格式

"rgb": "iVBORw0KGgoAAAANSUhEUgAAAOEAAADh...", RGB 图像数据，通常为 base64 编码格式

}

}

4.2.9 保存图片

save_base64_as_image（photo, file_path）

接口描述：将base64的图像数据，保存在指定路径下。

参数说明：

photo *: base64编码的字符串*
file_path: 保存图片的文件路径，包括文件名和扩展名

4.3 人脸识别

4.4 内容控制

4.4.1 文本转语音TTS（同步控制指令）

tts_sync(text)

接口描述：控制机器人调用tts算法，把文本转换为音频，并播报音频。

参数说明：

text，要转换为音频的文本内容

返回类型：

dict

返回说明：

{

"cmd": "response_text_audio",命令标识符，固定为 "response_text_audio"

"code": 返回状态码，0表示成功接收到请求；7010 表示完成tts音频播放；

"msg": 返回消息，通常为 "success"

"data": {}，可选的返回数据，本例中为空对象

}

文字转语音状态码：

状态码	枚举值	描述
0	START_TTS	开始文字转语音
7010	FINISHED_TTS	文字转语音结束

4.2.2 文本转语音TTS（异步控制指令）

tts(text)

接口描述：同上 4.2.1

参数说明：同上 4.2.1

返回类型：同上 4.2.1

返回说明：同上 4.2.1

状态码：同上 4.2.1

4.4.3 语音识别ASR（同步控制指令）

asr_sync(time)

接口描述：控制机器人调用asr算法，实时读取麦克风音频，从音频中识别文本，进而实现音频指令控制，默认录音5秒， 然后进行识别。

参数说明：

time, 录音的持续时间，范围从2秒到10秒，默认5s

返回类型：

dict

返回说明：

{

'cmd': 'response_audio_text', 命令标识符，固定为 "response_audio_text"

'code': 7000, 返回状态码，0表示成功接收到请求；7000 表示成功识别；7001 表示开始音频录制；7002 完成音频录制，开始进行语音识别

'msg': 'success',返回消息，通常为 "success"

'data': {'txt': '你好'}, 返回的数据，txt为返回数据的具体内容

}

语音识别状态码：

状态码	枚举值	描述

0	START_TTS	成功接收到请求
7010	SPEECH_RECOGNIZED	完成语音识别
7001	START_RECORDING	开始音频录制
7002	FINISH_RECORDING	完成音频录制，开始进行语音识别

4.4.4 语音识别ASR（异步控制指令）

asr(time)

接口描述：同上 4.4.3

参数说明：同上 4.4.3

返回类型：同上 4.4.3

返回说明：同上 4.4.3

状态码：同上 4.4.3

4.4.5 播放广告

show_media(type, url)

接口描述：控制机器人播放广告，支持图片、视频等格式文件。

参数说明：

type, 广告类型，必须是 "picture" 或 "video"

url，广告资源的URL地址

返回类型：

dict

返回说明：

{

"cmd": "response_play_ad",命令标识符，固定为 "response_play_ad"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": {}，可选的返回数据，本例中为空对象

}

4.5 上下桩

4.5.1 上桩（同步控制指令）

dock_charge_on_sync(map_id, x, y, theta)

接口描述：控制机器人上桩充电，

注意：机器人必须先设置地图，重定位成功，并且使用的地图设置有充电点位。

参数说明：

map_id：充电桩所在的地图编号

x: 充电桩的X坐标

y: 充电桩的Y坐标

theta: 上桩时的方向

返回类型：

dict

返回说明：

{

"cmd": "response_dock_ctrl",命令标识符，固定为 "response_dock_ctrl"

"code": 0,返回状态码，

"msg": 返回消息，上桩成功后是"0",

"data": {}，可选的返回数据，本例中为空对象

}

状态码

状态码	枚举值	描述
0	NAVI_TO_CHARGE_OK	完成导航到充电桩
14002	NAVI_TO_DOCK_START	开始导航到充电桩
14004	NAVI_TO_CHARGE	开始通过充电桩充电
14005	NAVI_TO_DOCK	正在导航到充电桩
6016	NAVI_NOPILEDETECTED	没有检测到充电桩

5.5.2上桩（异步控制指令）

dock_charge_on(map_id, x, y, theta)

接口描述：同上 4.5.1

参数说明：同上 4.5.1

返回类型：同上 4.5.1

返回说明：同上 4.5.1

状态码：同上 4.5.1

4.5.3 下桩（同步控制指令）

dock_charge_off_async()

接口描述：控制机器人下桩

参数说明：

无

返回类型：

dict

返回说明：

{

"cmd": "response_dock_ctrl", 命令标识符，固定为 "response_dock_ctrl"

"code": 0, 返回状态码

"msg": "success", 返回消息

"data": {},可选的返回数据，本例中为空对象

}

4.5.4 下桩（异步控制指令）

dock_charge_off()

接口描述：同上 4.5.3

参数说明：同上 4.5.3

返回类型：同上 4.5.3

返回说明：同上 4.5.3

4.6 通知

4.6.1 获取机器人当前位姿

get_location_pose()

接口描述：机器人心跳包，间隔10s上报一次，可以用来获取当前机器人在地图中的位置和姿态

参数说明：

无

返回类型

dict

返回说明

{

“cmd”:”notify_heart_beat”,命令标识符，固定为 "notify_heart_beat"

“code”:返回状态码，0表示成功

“msg”:返回消息，通常为 "success"

“data”：{包含系统心跳所需的数据

"x": 0, 当前位置的X坐标位置

"y": 0, 当前位置的Y坐标位置

"theta": 0, 当前位置的方向角度（以度为单位）

}

}

4.6.2 获取电池状态

get_battery_info()

接口描述：获取机器人电池信息，间隔10s上报一次

参数说明：

无

返回类型：

dict

返回说明：

{

“cmd”:”notify_battery_info”,命令标识符，固定为 "notify_battery_info"

“code”:0,返回状态码，0表示成功

“msg”:”success”,返回消息，通常为 "success"

“data”：{包含电池信息的数据

"battery": 剩余电量百分比，0-100

"status": 充电状态，0表示放电（未充电），1表示正在线充，2表示正在桩充

}

}

4.6.3 获取急停状态

get_emr_status()

接口描述：机器人急停状态上报

参数说明：

无

返回类型：

dict

返回说明：

{

“cmd”:”notify_emr_status”,命令标识符，固定为 "notify_emr_status"

“code”:返回状态码，0表示成功

“msg”:返回消息，通常为 "success"

“data”：{

"status": 紧急停止状态，0表示正常，1表示急停中

}

}

4.6.4 获取防撞条状态

get_collision_status()

接口描述：机器人防撞条状态上报

参数说明：

无

返回类型：

dict

返回说明：

{

“cmd”:”notify_collision_status”,命令标识符，固定为 "notify_collision_status"

“code”:返回状态码，0表示成功

“msg”:返回消息，通常为 "success"

“data”: {

"collisionLeft": 左侧碰撞状态，0表示未碰撞，1表示已碰撞

"collisionRight": 右侧碰撞状态，0表示未碰撞，1表示已碰撞

}

}

4.6.5 获取当前速度

get_speed()

接口描述：机器人实时速度上报

参数说明：

无

返回类型：

dict

返回说明：

{

“cmd”:”notify_chassis_speed”,命令标识符，固定为 "notify_chassis_speed"

“code”:返回状态码，0表示成功

“msg”:返回消息，通常为 "success"

“data”: {

"angleSpeed": 0, 角速度，单位通常是弧度/秒

"linearSpeed": 0 线速度，单位通常是米/秒

}

}

4.7 系统

设置音量

set_volume()

接口描述: 设置机器人扬声器音量

参数说明:

volume，音量值，范围从 0 到 100

返回类型:

dict

返回说明:

{

"cmd": "response_set_volume",命令标识符，固定为 "response_set_volume"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": {}，可选的返回数据，本例中为空对象

}

4.8 底盘控制

4.8.1 控制底盘运动

chassis_move(linear_speed,angle_speed,time)

接口描述: 控制机器人底盘运动

参数说明:

linearSpeed，直线速度，设置范围：0-0.5，单位：米/秒

angleSpeed，角速度，设置范围：0-0.5，单位：弧度/秒（注意单位与直线速度不同）

time，以当前设定的速度运动的运动时长，单位：秒

返回类型:

dict

返回说明:

{

"cmd": "response_chassis_move",命令标识符，固定为 "response_chassis_move"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": {}，可选的返回数据，本例中为空对象

}

4.8.2 停止底盘运动

chassis_stop()

接口描述：停止机器人底盘运动

参数说明：

无

返回类型：

dict

返回说明：

{

"cmd": "response_chassis_move",命令标识符，固定为 "response_chassis_move"

"code": 返回状态码，0表示成功

"msg": 返回消息，通常为 "success"

"data": {}，可选的返回数据，本例中为空对象

}

状态码	枚举值	描述
5000	MOVE_SUCCESS	底盘运动成功
5001	MOVE_CANCEL	底盘运动取消
5002	MOVE_FAILED	底盘运动失败