WATER Software API Manual

Document version: v1.8.7

Revision date: 2021-11-02

Parameters:

Interface description:

Example:

3. Get the current global status of the robot

Instructions:

Parameters:

Interface description:

Example:

Explanation of fields related to mobile tasks:

4. Get robot information interface

Instructions:

Parameters:

Instructions:

Example:

Explanation of returned result fields:

5. Marker function interface

5.1 Mark the marker at the current position of the robot

Instructions:

Instructions:

Parameters:

Interface description:

Example:

5.2 Get a list of marker points

Instructions:

Parameters:

Interface description:

Example:

5.3 Delete marker points

Instructions:

Parameters:

Interface description:

Example:

5.4 Get the number of points

Instructions:

Parameters:

Interface description:

Example:

5.5 Get summary information of points

Instructions:

Parameters:

Interface description:

Example:

5.6 Specify the coordinates to mark the marker

Instructions:

Parameters:

Interface description:

Example:

6. Direct robot control instructions

Instructions:

Parameters:

Interface description:

Example:

7. Robot emergency stop control instructions

Instructions:

Parameters:

Interface description:

Example:

8. Correct the current position of the robot

8.1 Specify the marker to correct the robot position

Instructions:

Parameters:

Interface description:

Example:

8.2 Specify coordinates to correct the robot position

Instructions:

Parameters:

Interface description:

Example:

9. Request real-time data from the robot

Instructions:

Interface description:

9.1 Request the real-time global status of the robot

Parameters:

Example:

9.2 Requester detects real-time data

Parameters:

Example:

9.3 Get the real-time speed of the robot

Parameters:

Example:

10. Proactive notification of robots

Interface description:

Notification list:

Field explanation:

Example:

Example:

11. Setting parameters

Instructions:

Parameters:

Interface description:

Example:

12. Get parameters

Instructions:

Parameters:

Interface description:

Example:

13. Wireless network interface

13.1 Get the list of WiFi currently available to the robot

Instructions:

Parameters:

Interface description:

Example:

13.2 Connect WiFi

Instructions:

Parameters:

Interface description:

Example:

13.3 Get the currently connected WiFi

Instructions:

Parameters:

Interface description:

Example:

13.4 Get robot IP and wireless network card address

Instructions:

Parameters:

Interface description:

Example:

13.5 Get detailed information about the wireless list currently available to the robot

Instructions:

Parameters:

Interface description:

Example:

14. Map interface

14.1 Get map list

Instructions:

Parameters:

Interface description:

Example:

14.2 Set the current map

Instructions:

Parameters:

Interface description:

Example:

14.3 Get the current map

Instructions:

Parameters:

Interface description:

Example:

14.4 Get map list details

Instructions:

Parameters:

Interface description:

Example:

14.5 Given a target point, find the location of a point near the target point

Instructions:

Parameters:

Interface description:

Example:

14.6 Given the target point, query the distance to the static map obstacle and the obstacle detected by the sensor

Instructions:

Parameters:

Interface description:

Example:

15. Shut down and restart interface

Instructions:

Parameters:

Interface description:

Example:

16. Software update interface

16.1 Get the current software version

Instructions:

Parameters:

Interface description:

Example:

16.2 Check for updates

Instructions:

Parameters:

Interface description:

Example:

16.3 Update

Instructions:

Parameters:

Interface description:

Example:

16.4 Restart service

Instructions:

Parameters:

Interface description:

Example:

17. Set the light strip interface

17.1 Set the brightness of the light strip

Instructions:

Parameters:

Interface description:

Example:

17.2 Set the color of the light strip

Instructions:

Parameters:

Interface description:

Example:

18. Self-diagnostic interface

18.1 Obtaining self-diagnostic results

Instructions:

Parameters:

Interface description:

Example:

19. Get power status interface

Instructions:

Parameters:

Interface description:

Example:

20. Get the robot global path interface

Instructions:

Parameters:

Interface description:

Example:

21. Get elevator status interface

Instructions:

Parameters:

Interface description:

Example:

22. Get the path interface between two points

Instructions:

Parameters:

Interface description:

Example:

a. Status code return description (for the status of the returned data)

- b. Error message return description (for error_message of returned data)

Important changes in each version:

v1.3.1 (2016-11-3)

- Add some explanatory text.
- Interface 3 Get the current global status of the robot. A field "move_retry_times" is added to feedback the number of internal retries during the robot's movement.
- Interface 5: Get the marker point list, modify three fields: name->marker_name, location->pose, type->key.
- v1.4.0 (2017-1-6)
- The following content has been modified for the software version 0.4.0:
- Remove the word http://192.168.10.10:8808 before the api command. (The original api format can still be used)
- Add "type" and other fields to the json sent by the server to distinguish packet types.
- Add an interface so that the server can send "robot status" at a certain frequency, see interface 9 to request real-time robot data.
- Add the function of actively sending status notifications on the server side, see interface 10 Robot Active Notification.
- v1.5.0 (2017-1-17)
- Add interface 11 to set parameters and interface 12 to get parameters, which can set/get robot parameters. (Currently only supports the robot maximum travel speed parameter)
- v1.6.0 (2017-5-29)
- In interface 11 setting parameters and interface 12 obtaining parameters, change the maximum travel speed of the robot to the percentage of the maximum travel speed of the robot.
- In interface 3, the current global state of the robot is acquired, and the soft and hard emergency stop state is added.
- Add interface 13 wireless network port, interface 14 map port, interface 15 shutdown and restart port.
- Interface 10 The robot proactively informs the interface of new elevator related notifications.
- v1.6.3 (2017-7-4)
- Increase interface 16 software update interface.
- v1.6.4 (2017-7-14)
- Update interface 15 Shut down and restart the interface.
- Interface 10 adds poweroff/emergency stop/charging status/recharge failure notification to the robot's active notification.
- v1.6.5 (2017-9-30)
- Add interface 17 to set the light strip interface.
- v1.6.7 (2017-10-19)
- Combine [Interface 4 insert new marker point] and [Interface 5 get marker point list] to interface 5 point (marker) function interface
- Added [Interface 5.3 delete marker point]
- Add interface 4 to obtain robot information
- v1.6.8 (2018-04-09)
- The following content has been added for the software version 0.7.3:

- Interface 1 Robot movement function. After the interface is successfully called, a field is added to the response to indicate the task ID.
- Interface 10 Active notification of robots, new descriptions of warnings when robots are trapped when they are moving and notifications of moving retry.
- Interface 10 Robots take the initiative to notify, and a new notification for robots to trigger posture correction is added.
- Interface 10 The robot proactively informs, if the new elevator space is not enough, it will be notified to take the next elevator.
- v1.7.1 (2018-05-29)
- The following content has been added for the software version 0.7.4:
- Interface 1 A cruise interface is added to the robot movement function.
- Interface 10 Robot active notification, new cruise notification, new gate/electronic door control notification. Added the data field as supplementary information for the notification.
- Interface 5.2 to get the list of marker points, add parameters to query points by floor. Add interface 5.4 to get the number of points, and add interface 5.5 to get point summary information.
- v1.7.2 (2018-07-10)
- The following content has been added for the software version 0.7.6:
- Interface 10 Active notification of robots, adding two types of notifications when robots are trapped. In the notification of the end of the task, a field is added to indicate the distance traveled by the task.
- v1.7.3 (2018-07-27)
- The following content has been added for the software version 0.7.7:
- In interface 3, the "error_code" field is added to the status returned by obtaining the current global status of the robot.
- Interface 9 requests the robot real-time data to add a human detection data interface.
- Added interface 13.5 to obtain detailed information about the wireless list currently available to the robot.
- v1.7.4 (2018-08-17)
- The following content has been added for the software version 0.7.8:
- Interface 14.3 to obtain information such as increased map resolution in the current map.
- v1.7.5 (2018-09-04)
- The following content has been added for the software version 0.7.9:
- Add interface 19 to obtain the power status interface.
- Interface 10 Robots take the initiative to notify, and the notification of the failure of global path planning is added.
- v1.7.6 (2018-09-07)
- Modify the value range of the robot travel speed proportional parameter in the interface 11 setting parameters, and add the description of the parameters.
- v1.7.7 (2018-11-15)
- Added 9.3 to obtain the robot real-time speed interface.
- Added 5.6 designated coordinate marking marker interface.

- Added 8.2 designated coordinate correction robot position interface.
- v1.7.8 (2019-02-19)
- The maximum number of consecutive retries parameter is added to the interface 1 robot movement function.
- Interface 1.1 The distance and angle tolerance parameters are added in the movement of a single target point.
- In interface 3, the "running_status" field is added to the current global status of the robot.
- v1.7.9 (2019-06-15)
- Interface 15 Shutdown restart interface adds the parameter setting delay restart time.
- In interface 10, a notification of waiting for the elevator to be unlocked is added to the active notification of the robot.
- Interface 3 Get the current global status of the robot. The running_status field adds the status of waiting for the elevator to be unlocked.
- v1.8.0 (2019-09-16)
- In the interface 1.1 single target point movement, the angle offset parameter is added.
- Interface 10 adds a notification that the software service is about to be closed in the active notification of the robot.
- v1.8.1 (2019-10-26)
- Added 16.4 restart service interface.
- Add 20 to get the robot global path interface
- v1.8.2 (2020-03-16)
- Added 21 interface for obtaining elevator status
- Update the interface 11 setting parameters, and add the interface for setting the maximum linear velocity and angular velocity of the robot.
- v1.8.3 (2020-07-16)
- Interface 1.1 adds the parameter of whether to allow two-way docking during the movement of a single target point, which is only valid for robots that support two-way walking.
- Interface 1.1 The parameter of the concession stop distance is added in the movement of a single target point to improve the flexibility of the robot to complete tasks in certain scenarios.
- Add 17.2 to set the light strip color interface.
- v1.8.4 (2021-02-23)
- Added 22. Get the path interface between two points.
- In interface 10, the robot active notification adds a notification that the mobile task directly fails due to the lack of an available path.
- v1.8.5 (2021-03-11)
- Added 14.4 interface for obtaining map list details.
- v1.8.6 (2021-05-19)
- Interface 10 The notification of "the robot may get lost" is added to the active notification of the robot.
- In interface 10, the notification of "recharge timeout failed" is added to the active notification of the robot.

- v1.8.7 (2021-11-02)
- Increase interface 14.5 to give the target point, find the position of the reachable point near the target point
- Add interface 14.6. Given a target point, query the distance between obstacles on the static map and the obstacles detected by sensors

Applicable environment:

- Network communication based on TCP socket.

Protocol form and connection method:

Protocol form: The interface content adopts the url-like string format, and the interface feedback data adopts the **json** format uniformly. There is **no** context between fields at the same level. Please use the standard json parsing method when parsing.

Connection method: TCP connection

Network settings: TCP client

Server IP address:

a) 192.168.10.10 (this is the static IP address of the chassis host, the TCP client host needs to establish a LAN connection with the chassis host through a router and other devices, and it has the same network segment 192.168.10.*)

b) The chassis is connected to the LAN WiFi through API, and then the LAN IP obtained through API.

Server port: 31001

It is recommended to use the socket connection method, because serial communication has a certain probability of data transmission errors.

Interface Instructions:

General explanation:

O Universal parameter "uuid"

o If no special instructions are given, uuid can be carried as an optional parameter in the following commands, and the server will return it as it is in the **response** after the command is executed.

Explanation of the "type" field in json:

o "response" means a response to an instruction

o "callback" means data sent by the server at a certain frequency

o "notification" means an active notification initiated by the server

Example:

The client sends a string: "/api/move?marker=target_name&uuid=123456"

return:

```
{
    "type": "response", // package type
    "command": "/api/move", // source command
    "error_message": "", // error message
    "status": "OK", // execution result
    "uuid": "123456" // user-defined uuid
}
```

1.Robot movement function

1.1Single target point movement

Instruction:

```
/api/move
```

Parameter:

Name	Illustration	Is it mandatory	Remark	From the following version
marker	Target point code	And location must choose one	Priority is lower than location	

Name	Illustration	Is it mandatory	Remark	From the following version
location	x<the x-axis coordinate in the map>, y<the y-axis coordinate in the map>, theta<the relative theta value in the map>	Must choose one with marker		
max_continuous_retries	The maximum number of continuous retries in place (when the robot is not moving, the task will fail if the number of retries exceeds this value)	Optional	30 times by default	0.7.11
distance_tolerance	Distance tolerance, type float, unit meter. (When the target position is occupied and cannot be reached due to reasons such as, the task is considered successful if the robot moves within this distance of the target.)	Optional	The default value is related to the robot model	0.7.12
theta_tolerance	Angle tolerance, type float, unit radian. (After reaching the target point, the mission is successful when the angle is less than this value)	Optional	The default value is related to the robot model	0.7.12
angle_offset	The angle offset after reaching the position, for example, when using marker=m1 to send a task, the task will be executed with the angle of m1 + angle_offset as the final direction.	Optional	Unit radian, range [-3.14, 3.14], default 0	0.8.2
yaw_goal_reverse_allowed	**Two-way** parking control parameter, the value is 1 or 0 or -1. For a bidirectional robot, this parameter is used to allow the tail to be in the same direction as the point when the robot stops at a point.	Optional	1: allow; 0: not allow; others: use the default	0.8.7
occupied_tolerance	The parameter for the distance of the concession stop, in meters. When the target point is occupied, the robot will directly stop near the point to complete the task by setting this parameter, instead of trying to move to the point.	Optional	[0.1,)	0.8.7

Name	Illustration	Is it mandatory	Remark	From the following version
	The distance is calculated from the edge of the occupant to the center of the robot.			

Interface Description:

The robot can be moved from the current position to a target point that has been calibrated on the map.

The agreement includes two main forms for the move instruction part:

- For those who need to make the robot use automatic navigation and obstacle avoidance functions during the movement, please use this interface;
- If you need to use remote control functions such as handles, please use interface 6, which will directly control the speed of the two wheels of the robot;

Before using this interface, there is still necessary scanning work that needs to be arranged by the user in advance (Yunji provides a solution to connect to the robot through the LAN, and then log in to the browser to scan the image. For specific operations, please refer to the "Water User Manual").
The coordinates of the location in the parameter are relative to the "map" coordinate system. Because the direct call (x, y, theta) is not intuitive and convenient in use, the cloud trail provides "pre-mark anchor points, and then make the robot return to the anchor point." " Work form. Afterwards, the "anchor point" is collectively referred to as marker, and the method of setting and querying the anchor point is shown in interface 4 and interface 5.

When the robot adopts automatic navigation and obstacle avoidance, the robot will automatically plan the path and adjust the speed without operator intervention. The operator can monitor robot_status (robot overall and travel status), or wait for the corresponding notification to understand the task completion and the status of the whole machine.

Example:

Send:

```
/api/move?marker=target_name
// Call the mobile interface and move to the target point codenamed "target_name"
/api/move?location=15.0,4.0,1.5707963
// Call the mobile interface to move to the target point of location (15.0, 4.0, pai/2)
```

Return:

```
{
```

```
   "type": "response",
   "command": "/api/move",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "task_id": "xxx" // (Newly added after software version 0.7.3) 32-byte uuid,
for example:436253D1D6284ACC984620CCB94EB5E5}
```

1.2 Multi-target movement

Instruction:

```
/api/move
```

Parameter:

Name	Illustration	Is it mandatory	Note	From the following version
markers	A list of points you want to cruise, separate the point names with English commas	Required	No less than two points, no cross-floor support	
count	The number of cruises, all points are counted as one after going through	Optional	When not selected, the default value is once, and -1 means infinite loop.	
distance_tolerance	The tolerance when the point is reached, indicates how close to the point to continue to the next point.	Optional	Default 0.5 (m), minimum 0.5	
max_continuous_retries	Maximum number of continuous retries in place (when the robot is not moving in place, if the number of retries exceeds this value, the mission to the **current** cruise point will fail)	Optional	5 times by default	0.7.11

Interface Description:

As an extension of interface 1.1 single target point movement, it realizes **uninterrupted** cyclic movement among multiple points. Considering the case of starting from the charging pile, it does not support movement between multiple floors. After the v0.7.11 version, it supports returning to the charging pile.
The mission status is "running" during the entire cruise, cruise notifications will be issued at the beginning and end of the cruise, and general movement notifications will be issued between points. After a single cruise target point fails to move, it will continue to the next target point.
The command to end the cruise is the same as the command to end the single-point movement task, both through the interface 2 movement cancellation function.

Example:

Send:

```
/api/move?markers=m1,m2,m3&distance_tolerance=1.0&count=-1
```

Return:

```
{
   "type": "response",
   "command": "/api/move",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "task_id": "xxx" // (Newly added after software version 0.7.3) 32-byte uuid,
for example:436253D1D6284ACC984620CCB94EB5E5}
```

2. Mobile cancellation function

Instruction:

```
/api/move/cancel
```

Parameter:

N/A

Interface description:

Make the robot actively abandon the currently executing mobile task, and after successful cancellation, the robot can enter a new standby state.

When the robot executes the interface 1-robot movement command, if the current movement state of the robot needs to be terminated, this interface can be called. After receiving the "move cancel" command, the robot will stop in place and wait for another move command.

Example:

Send:

```
/api/move/cancel
// Cancel the move instruction currently in progress
```

Return:

```
{
    "type": "response",
    "command": "/api/move/cancel",
    "uuid": "",
    "status": "OK",
    "error_message": ""}
```

3. Get the current global status of the robot

This interface is reserved for the previous software version compatible with users. New users are recommended to use interface 9-request real-time data of the robot.。

Instruction:

```
/api/robot_status
```

Parameter:

N/A

Interface Description:

Get the current global status of the robot, including the status of the mobile task.

In addition to monitoring the feedback of the move in one cycle with the interface 1, you can also monitor other information of the robot from this interface, including "whether it is in a charging state", "whether it is in an emergency stop state", "the percentage of battery points left", "Current position relative to the map", "Current floor", see the example for details.

It is recommended that the calling frequency is 1-2HZ, and the task status can be monitored in real time as a logical judgment for process control.

Example:

Send:

```
/api/robot_status
// Get the current global status of the robot
```

Return on success:

```
{
   "type": "response",
   "command": "/api/robot_status",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {
      "move_target": "target_name", // The name of the target point specified by the move command
      "move_status": "running", // The execution status of the move instruction. See below for detailed explanation
      "running_status": "running", // New in v0.7.12, the specific status of the mobile task, see the following explanation for details
      "move_retry_times": 3, //Each time this number of times increases by 1, it means that the robot has made a new round of path retry; the path planning is successful at one time, this value defaults to 0

      "charge_state": bool, //true->Charging state. false->Uncharged state.
      "soft_estop_state": bool, // Soft emergency stop status set via API interface, true->emergency stop, false->non-emergency stop
      "hard_estop_state": bool, // The hard emergency stop state set by the hardware emergency stop button, true->emergency stop, false->non-emergency stop
      "estop_state": bool,  // hard_estop_state || sofpt_estop_state, true->Emergency stop, false->non-emergency stop
      "power_percent": 100, //Percentage of electricity, unit:%
      "current_pose": {
         "x": 11.0,    // Unit: m
         "y": 11.0,      // Unit: m
         "theta": 0.5, //Unit: rad
      }
      "current_floor": 16,
```

```
      "chargepile_id": "1234", // Added in v0.9.6. In the charging state, it indicates the ID of the charging pile currently being charged, and it returns "0" in the non-charging state. Note: This field is only valid in some products, and the rest return "0".
      "error_code": "00000000"   // New in v0.7.7, hexadecimal error code, a total of 8 bytes, non-zero means the robot is abnormal
   }
}
```

Return on failure:

```
{
   "type": "response",
   "command": "/api/robot_status",
   "uuid": "",
   "status": "UNKNOWN_ERROR",
   "error_message": "Can't catch current robot status"
   "results"
}
When it fails, please check whether the map is set correctly or whether the hardware is faulty.
```

Explanation of fields related to mobile tasks:

The following fields correspond to the status query of the robot movement function command.

-

move_target Indicates the point name specified by the movement command.
When calling the mobile interface with "location", the value of this field is empty

When calling the cruise interface, this field is the name of the current point

-
    -

move_status represents the execution status of the current robot to **move_target**. . Its value and explanation are as follows.。

-

Field value	Explanation

Field value	Explanation
idle	Indicates that the robot service has not received any movement instructions since it started.
running	Indicates that the robot is going to explain **move_target**, and will refuse to accept new move instructions at this time.
succeeded	Indicates that the mobile task has been successfully completed.
failed	Indicates that the move task has failed.
canceled	Indicates that the move task has been cancelled.

-
-

running_status represents the specific status of the current robot mobile task. The underlying implementation is triggered by notifications. In the case that the notification cannot be guaranteed to be received, this field can be used as a judgment condition.

-

Field value	Corresponding to move_status	Explanation	From the following version	Special Note
idle	idle/suceeded/failed/canceld	The robot is currently idle and can accept new mobile tasks.	v0.7.12	
leave_charging_pile	running	Leaving the charging pile.	v0.7.12	

Field value	Corresponding to move_status	Explanation	From the following version	Special Note
dock_to_charging_pile	running	It is automatically docking on the charging pile.	v0.7.12	
goto_lift	running	Going to the elevator.	v0.7.12	
wait_lift_unlock	running	Wait for the elevator to unlock.	v0.8.1	This state is triggered when multiple robots need to take the same elevator.
wait_lift_outside	running	Waiting for the elevator outside the elevator.	v0.7.12	

Field value	Corresponding to move_status	Explanation	From the following version	Special Note
enter_lift	running	Entering the elevator.	v0.7.12	It is best not to call the mobile cancel function casually at this time, In order to avoid process errors.
avoid_lift	running	After failing to enter the elevator, he is avoiding the elevator.	v0.7.12	It is best not to call the mobile cancel function

Field value	Corresponding to move_status	Explanation	From the following version	Special Note
				casually at this time, In order to avoid process errors.
take_lift	running	Taking the elevator.	v0.7.12	It is best not to call the mobile cancel function casually at this time, In order to avoid process error

Field value	Corresponding to move_status	Explanation	From the following version	Special Note
				s.
exit_lift	running	Getting out of the elevator.	v0.7.12	It is best not to call the mobile cancel function casually at this time, In order to avoid process errors.
back_to_lift	running	Failure to exit the elevator is returning to the elevator.	v0.7.12	It is best not to call the mobile cancel funct

Field value	Corresponding to move_status	Explanation	From the following version	Special Note
				ion casually at this time, In order to avoid process errors.
running	running	Other non-critical status	v0.7.12	

-

4. Get robot information interface

This interface requires at least software version 0.6.3.3

Instruction:

```
/api/robot_info
```

Paramater:

N/A

Interface description:

Some basic information of the robot can be obtained by calling the interface.

Example:

Send:

```
/api/robot_info
```

Return:

```
{
   "type": "response",
   "command": "/api/robot_info",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {
      "product_id": "WATER-xxxx-xxxxx"  // Robot number
    }}
```

Return result field explanation:

Result field	Explanation	Remark	From the following version
product_id	Product number		0.6.3.3

5.Marker Function interface

5.1Mark at the current position of the robot-marker

Instruction:

```
/api/markers/insert
```

Instruction Descriptions:

Mark anchor points at the current position and floor of the robot(marker)。

Parameter:

Name	Illustration	Is it Mandatory	Note

Name	Illustration	Is it Mandatory	Note
name	Point name (string type, special characters are not supported)	required	If name already exists, the coordinates are updated.
type	Point type (int type)	Optional, default is 0	Common types are: 0 (general point), 1 (front desk), 7 (gate), 3 (outside the elevator), 4 (in the elevator), 11 (charging pile) and so on. Each type of point has a specific function and its own attributes. Except for the common types, please use the robot monitoring page to add as much as possible to avoid abnormalities in the operation of the program due to abnormal attribute values. Under normal circumstances, user-defined types are not recommended. If you use a custom type, please use a value greater than 1000 to avoid conflicts with the type defined by the robot. As the robot software version is different, new types and corresponding attributes may be added in the future. Please refer to the marked points in the monitoring page mapping tool.
num	Point number (int type)	Optional, default is 1	Certain types of points have num (number) attributes, such as elevator points, gate points, charging pile points, etc.。

Interface Description:

Mark a point on the current map and let the robot remember this point (marker).

When calling this interface, the robot will capture the current (x, y, theta) relative to the map coordinate system, and automatically store the captured anchor point position in the robot's anchor point list to respond to the move call at any time. If the charging function is not used, the type in the interface need not be set.

Example:

Send:

```
/api/markers/insert?name=205_room
//Call this interface and set the current position of the robot as the marker point, with the name 205_room. The default is level 1 general points/api/markers/insert?name=charge_dock_2&type=11
//Call this interface and set the current position of the robot as the marker point, the name is charge_dock_2, and the type is charging pile.
```

Return:

```
Return when the operation is successful:
{
   "type": "response",
   "command": "/api/markers/insert",
   "uuid": "",
   "status": "OK",
   "error_message": ""
}
```

5.2 Get a list of marker points

Instruction:

```
/api/markers/query_list
```

Parameter:

Name	Illustration	Is it mandatory	Remark	From the following version
floor	Floor which want to query	Optional	If not selected, it will return the point information of all floors	0.7.4

Interface Description:

Get all the marker information of the robot in the current map. Each point information includes "point name", "floor", "point coordinate", "point direction" and "point type".
The orientation saves the direction of the point in the form of a quaternion, which can be transformed into the theta in the coordinate system described above

Example:

Send:

```
/api/markers/query_list
```

Return:

```
{
   "type": "response",
   "command": "/api/markers/query_list",
   "uuid": "",
```

```
   "status": "OK",
   "error_message": "",
   "results":{
      "meeting_room": {
         "floor": 1,
         "pose": {
            "orientation": {
               "w": -0.899999976158142,
               "x": 0,
               "y": 0,
               "z": -0.430000007152557
            },
            "position": {
               "x": -8.57999992370605,
               "y": 6.3600001335144,
               "z": 0
            }
         },
         "marker_name": "meeting_room",
         "key": 0
      },
      ... // Other point information
   }
}
// "marker_name" represents the name of the point. "key" indicates the point type: 0 (general point), 11 (charging pile). "floor" indicates the floor to which the point belongs. "pose" represents the specific coordinate point.
```

Return when the point does not exist:

```
{
   "type": "response",
   "command": "/api/markers/query_list",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": null}
```

5.3 Delete marker points

Instruction:

```
/api/markers/delete
```

Parameter:

Name	Illustration	Is it mandatory	Remark	From the following version
name	Point name	Required		0.6.3.4

Interface description:

Delete the marked marker point. If the point name does not exist, it will return to failure.

Example:

Send:

```
/api/markers/delete?name=205_room
```

Return on success:

```
{
    "type": "response",
    "command": "/api/markers/delete",
    "uuid": "",
    "status": "OK",
    "error_message": ""}
```

Return on failure:

```
{
    "type": "response",
    "command": "/api/markers/delete",
    "uuid": "",
    "status": "INVALID_REQUEST",
    "error_message": "Marker Not Found"}
```

5.4 Get the number of points

This interface requires at least software version v0.7.4

Instruction:

```
/api/markers/count
```

Parameter:

N/A

Interface description:

Get the number of points in the current map

Example:

Send:

```
/api/markers/count
```

Return:

```
{
   "type": "response",
   "command": "/api/markers/count",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {
     "count": 10  // The number of points in the current map
   }}
```

5.5 Obtain summary information of points

This interface requires at least software version v0.7.4

Instruction:

```
/api/markers/query_brief
```

Parameter:

N/A

Interface description:

Query the summary information of all points on the current map. This interface returns more concise point information than 5.2 obtaining the list of marker points

Example:

Send:

```
/api/markers/query_brief
```

Return:

```
{
   "type": "response",
   "command": "/api/markers/query_brief",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {
     "meeting_room": "0-1", // Point name: point type-floor
     "205_room": "0-1",
     ... // Other point information
    }
}
```

5.6 Specify the coordinate mark-marker

This interface requires at least software version v0.7.11

Instruction:

```
/api/markers/insert_by_pose
```

Parameter:

Name	Illustration	Is it Mandatory	Remark
name	Point name (string type, special characters are not supported)	Required	If name already exists, the coordinates are updated.
type	Point type (int	Optional,	Common types are: 0 (general point), 1 (front desk), 7 (gate), 3

Name	Illustration	Is it Mandatory	Remark
	type)	default is 0	(outside the elevator), 4 (in the elevator), 11 (charging pile) and so on. Each type of point has a specific function and its own attributes. Except for the common types, please use the robot monitoring page to add as much as possible to avoid abnormalities in the operation of the program due to abnormal attribute values. Under normal circumstances, user-defined types are not recommended. If you use a custom type, please use a value greater than 1000 to avoid conflicts with the type defined by the robot. As the robot software version is different, new types and corresponding attributes may be added in the future. Please refer to the marked points in the monitoring page mapping tool.
num	Point number (int type)	Optional, default is 1	Certain types of points have num (number) attributes, such as elevator points, gate points, charging pile points, etc.。
floor	Floor (int type is not 0)	Optional	The default is the current floor of the robot, and an error will be returned if the floor does not exist.
x	Map coordinate x (float type)	required	-
y	Map coordinate y (float type)	required	-
theta	Point direction (float type)	required	Value range [-π, π]

C++ example of theta and quaternion conversion relationship:

Suppose the direction represented by the quaternion is: o(x: 0, y:0, z: -0.430000007152557, w: -0.899999976158142)

```
double theta = 2*atan2(o.z, o.w)
if (theta > M_PI && theta <= (2*M_PI))
{
   theta -= 2*M_PI;
}
else if(theta < (-1*M_PI) && theta >= (-2*M_PI))
{
   theta += 2*M_PI;
}
```

Interface Description：

Add a marker at the specified location.

Example:

Send:

```
/api/markers/insert_by_pose?name=205_room&x=-0.1&y=1.0&theta=0.0&floor=2&type=0
//Call this interface and add a marker named 205_room at the location (0.1, 1.0), direction (0.0), and floor 2 location, and the type is general point.
```

Return:

```
Return when the operation is successful:
{
   "type": "response",
   "command": "/api/markers/insert_by_pose",
   "uuid": "",
   "status": "OK",
   "error_message": ""
}
```

6.Direct robot control instructions

Instruction:

```
/api/joy_control
```

Parameter:

Name	Illustration	Is it Mandatory	Remark
angular_velocity	Angular velocity	Required	
linear_velocity	Line speed	Required	

Interface Description:

Part of the direct control of the robot, such as autorotation or stop (this control priority is higher than the move instruction), and return to successful indicates that the robot has successfully received and started to run this instruction

> Interface 1 has provided the robot with path planning and automatic obstacle avoidance movement functions. Relatively speaking, this interface provides manual control of remote control movement functions. The method of remote control is to control the linear velocity and

angular velocity of the robot, and the units are m/s and rad/s; through the two-value setting, you can control the robot movement manifestations such as the turning radius by yourself.

The robot movement direction, angular velocity and linear velocity symbols are explained as follows:

Line speed	Angular velocity	Robot performance	Remark
0	Positive	The robot turns left on the spot	The robot angular velocity setting range is (-1.0 ~ 1.0)rad/s
0	Negative	The robot turns right on the spot	
Positive	0	Robot advance	The range of robot linear velocity is (-0.5 ~ 0.5)m/s
Negative	0	Robot back	
Positive	Positive	The robot moves forward and turns left	
Positive	Negative	The robot moves forward and turns right	
Negative	Positive	The robot moves forward and turns left	
Negative	Negative	The robot moves forward and turns right	

If the incoming parameters exceed the limited range, the module will work according to the maximum preset

For safety reasons, the duration of a single command is 0.5 seconds, but the input frequency can be greater than 2 Hz. Continuous command sending can make the robot move coherently, such as mapping commands to handle control keys

Example:

Send:

```
/api/joy_control?angular_velocity=0.5&linear_velocity=0.2
// Call the mobile interface, the robot rotates counterclockwise at an angular velocity of 0.5rad/s, and moves forward at a linear velocity of 0.2m/s at the same time
```

Return:

```
{
   "type": "response",
   "command": "/api/joy_control",
   "uuid": "",
   "status": "OK",
   "error_message": ""}
```

7. Robot emergency stop control instructions

Instruction:

```
/api/estop
```

Parameter:

名称	说明	是否必选	取值范围	备注
flag	进入或退出急停模式	必选	true/false	

Interface Description:

Make the robot enter the free stop mode, the robot can be pushed.

> **The robot hardware platform provides an emergency stop control button. Press this button to make the robot stop automatically. In addition to this hardware button, the software interface provides an equivalent emergency stop control command. After the software starts the emergency stop mode, the robot can also enter the free stop mode. It should be noted that the software and hardware emergency stop interfaces control the emergency stop state of the robot separately, and cannot unlock each other.**

Note: After the software restarts or the whole machine restarts, the emergency stop status will be reset to False (after v0.6.4)

Example:

Send:

```
/api/estop?flag=true  //Enter emergency stop mode
/api/estop?flag=false  //Exit emergency stop mode
```

返回:

```
{
   "type": "response",
   "command": "/api/estop",
   "uuid": "",
   "status": "OK",
   "error_message": ""}
```

8. Correct the current position of the robot

When the position of the robot has shifted in the map, in order to ensure the stable operation of the robot, it is necessary to help the robot to correct the position.

Note: This type of interface is generally not used in the development process, especially in the process of moving tasks, there is no need to call this interface to correct the position. Under normal circumstances, you can use the correction function provided by the front end

8.1Specify the marker to correct the robot position

Instruction:

```
/api/position_adjust
```

Parameter:

Name	Illustration	Is it mandatory	Ranges	Remark
marker	The name of the marker used to calibrate the position	Required	Marker points that have been calibrated	

Interface Description:

Use this interface to correct the robot position to the position marked by the marker. When using, push the robot to the position marked by the marker first, and then use this interface to perform position correction.

Example:

Send:

```
/api/position_adjust?marker=001
//Tell the robot that it is currently at the marker point code-named 001
```

Return:

```
{
   "type": "response",
   "command": "/api/position_adjust",
   "uuid": "",
   "status": "OK",
   "error_message": ""}
```

8.2 Specify the coordinates to correct the robot position

This interface requires at least software version v0.7.11

Instruction:

```
/api/position_adjust_by_pose
```

Parameter:

Name	Illustration	Is it mandatory	Ranges	Remark
x	Coordinate x	Required		
y	Coordinate y	Required		
theta	Direction	Required		
floor	floor	Optional	Floors that exist in the current map	If left blank, it will default to the current floor of the robot

Interface Description:

Use this interface to correct the robot position to the specified coordinate position.

Example:

Send:

```
/api/position_adjust_by_pose?x=0.3&y=0.3&theta=2.6
// Correct the robot position to the specified coordinates
```

返回:

```
{
   "type": "response",
   "command": "/api/position_adjust_by_pose",
   "uuid": "",
   "status": "OK",
   "error_message": ""}
```

9. Request real-time data from the robot

Instruction:

```
/api/request_data
```

Interface Description:

Request the server to send topic type data at a certain frequency. When the request is successful, the server will send data to the requesting client at a certain frequency.

9.1 Request the real-time global status of the robot

This interface requires at least software version v0.4.0

Parameter:

Name	Illustration	Is it mandatory	Ranges	Remark	From the following version
topic	Type of real-time data requested	Required	robot_status		v0.4.0

Name	Illustration	Is it mandatory	Ranges	Remark	From the following version
switch	Start/stop data sending	Required	on/off	Deprecated	v0.4.0-v0.7.6.3
frequency	Sending frequency	Optional	>0.0	Default 2HZ	v0.4.0

Example:

Send request:

```
/api/request_data?topic=robot_status&frequency=1
// Request the server to send the global status of the robot at a frequency of 1HZ
```

Return:

```
{
   "type": "response",
   "command": "/api/request_data",
   "uuid": "",
   "status": "OK",
   "error_message": ""}
```

After a successful request, the server sends data with topic robot_status at a frequency of 1HZ, and type as callback

```
{
   "type": "callback",
   "topic": "robot_status",
   "results":
   {
   ...   // See the content and explanation of "results" in interface 3 for details
   }
}
```

9.2 Requester detects real-time data

This interface requires at least software version v0.7.7

Note: This interface needs to configure and enable the human leg recognition module

Parameter:

Name	Illustration	Is it mandatory	Range	Remark	From the following version
topic	Type of real-time data requested	Required	human_detection		v0.7.7
frequency	Sending frequency	Optional	>0.0	Default 1HZ	v0.7.7

Example:

Send request:

```
/api/request_data?topic=human_detection&frequency=0.5
// Request the server to send human detection results at a frequency of 0.5HZ
```

返回:

```
{
   "type": "response",
   "command": "/api/request_data",
   "uuid": "",
   "status": "OK",
   "error_message": ""}
```

After a successful request, the server sends **topic** as human_detection and type as **callback** at a frequency of 0.5HZ

```
{
   "type": "callback",
   "topic": "human_detection",
   "results": {
      "legtrack 0": {   // Identification ID
         "position": {  // Position, the front of the robot is the positive x-axis,
the right-hand coordinate system
            "x": -0.4504653354557998,
            "y": -0.6021136068883720
         },
         "velocity": { // Speed
            "x": 0.0002109676660555934,
            "y": :0.0001660708499816679
         }
      },
      "legtrack 1": {
         "position": {
            "x": -0.4504653354557998,
```

```
            "y": -0.6021136068883720
        },
        "velocity": {
            "x": 0.0002109676660555934,
            "y": :0.0001660708499816679
        }
    }
  }
}
```

9.3 Get the real-time speed of the robot

This interface requires at least software version v0.7.10

Parameter:

Name	Illustration	Is it mandatory	Range	Remark	From the following version
topic	Type of real-time data requested	Required	robot_velocity		v0.7.10
frequency	Sending frequency	Optional	>0.0	Default1HZ	v0.7.10

Example:

Send request:

```
/api/request_data?topic=robot_velocity&frequency=0.5
// Request the server to send the current speed of the robot at a frequency of 0.5HZ
```

Return:

```
{
   "type": "response",
   "command": "/api/request_data",
   "uuid": "",
   "status": "OK",
   "error_message": ""}
```

After a successful request, the server sends **topic** as robot_velocity and type as **callback** at a frequency of 0.5HZ

```
{
   "type": "callback",
```

```
   "topic": "robot_velocity",
   "results": {
      "angular": 0.1,   // Robot angular velocity, greater than 0 means left turn, less than 0 means right turn
      "linear": 0.1    // Robot linear speed, greater than 0 means forward, less than 0 means backward
   }
}
```

10. Proactive notification of the robot

Interface Description:

The robot actively sends a notification to all connected clients, and the client can perform corresponding actions after receiving the notification, such as voice broadcast.

Note: It is not recommended as a logical judgment for process control, as the notification may not be received due to network and other reasons. It is recommended to obtain the current global state of the robot through interface 3 to obtain the real-time state as the logical judgment of the process.

Notification list:

code	description	Chinese explanation	level	From the following version	Illustration
General mobile tasks:					Need to call interface 1 robot movement function to trigger the following notification
“01001”	The move task is started.	移动任务开始了。	info	0.4.0	
“01002”	The move task is finished.	移动任务完成了。	info	0.4.0	
“01003”	The move task is failed.	移动任务失败了。	warning	0.4.0	After **continuous** movement retry (01004) for a certain number of times, the robot has hardly moved, this notification will be issued and the task will be judged to have failed.
“01004”	The move task is canceled.	移动任务被取消了。	info	0.4.0	Call interface 2 After the move cancel function successfully

code	description	Chinese explanation	level	From the following version	Illustration
					cancels the move task, this notification will be sent.
“01005”	The move task is retried.	移动重试。	info	0.4.0	Every time the robot attempts to move and fails, and does not meet the condition of moving failure (01003), this notification will be issued, and then it will re-plan the path to the target point.
“01006”	The robot may be trapped.	机器人可能被困住了。	warning	0.7.3	This notification will be issued after a certain number of **continuous** movement retries (01004) and the robot has barely moved. At this time, human intervention may be required to help the robot get out of the trap.
“01007”	Failed to find available path.	找不到可用的路径。	warning	0.9.6	Before the start of each movement task, if the robot judges that the current position cannot reach the target position, it will send this notification directly, and then send the “01003” movement task failure notification, and no more movement retry. This notification is triggered at most for each movement task Once. This notification is triggered for the following reasons: the current position of the robot or the task target is in an area that cannot be moved (the gray area on the map, the black line on the map, outside the map), or the elevator cannot be found when it is necessary to take the elevator.
“01010”	Start to leave charging pile.	开始离开充电桩。	info	0.5.2	
“01011”	Succeed to	离开充电桩成功	info	0.5.2	

code	description	Chinese explanation	level	From the following version	Illustration
	leave charging pile.	了。			
“01012”	Failed to leave charging plie.	离开充电桩失败了。	warning	0.5.2	
“01020”	Start to auto dock to charging pile.	开始自动停靠到充电桩。	info	0.5.2	
“01021”	Succced to auto dock to charging pile.	自动停靠到充电桩上成功了。	info	0.5.2	
“01022”	Failed to auto dock to charging pile.	自动停靠到充电桩上失败了。	warning	0.5.2	
“01023”	Failed to to receive valid data.	回充失败，数据异常。	warning	0.5.10	
“01024”	Failed to find any feature around the robot.	回充失败,没找到充电桩。	warning	0.5.10	
“01025”	Failed to catch power status.	回充失败，没收到上电信号。	warning	0.5.10	
“01026”	Failed to catch infrared signal.	回充失败，没收到红外信号。	warning	0.5.10	
“01027”	Failed to dock with timeout.	回充失败,超时。	warning	0.9.8	
“01030”	Start controlling the electronic door.	开始控制电子门/闸机。	info	0.7.4	
“01031”	Finish controlling the electronic door.	结束控制电子门/闸机。	info	0.7.4	

code	description	Chinese explanation	level	From the following version	Illustration
“01032”	Control the electronic door for too long, forced release control.	控制电子门/闸机已经超时，强制释放控制。	warning	0.7.4	
“01101”	The cruise is started.	巡游任务开始了。	info	0.7.4	
“01102”	The cruise is finished.	巡游任务完成了。	info	0.7.4	
“01103”	The cruise is failed.	巡游任务失败了。	warning	0.7.4	Send this notification when the last mobile task fails
“01104”	The cruise is canceled.	巡游任务被取消了。	info	0.7.4	This notification will be issued after calling interface 2 after the movement cancellation function successfully cancels the cruise task.
“01201”	Trapped in unknown area.	被困在未知区域(地图中灰色区域)	warning	0.7.6	
“01202”	Trapped in obstacle.	被困在障碍物附近(地图中黑线)	warning	0.7.6	
“01203”	Failed to make global plan.	全局路径规划失败(困在灰色区域、被挡死时大概每 5 s 触发一次)	info	0.7.9	
Elevator Task:					Need to call interface 1 robot movement function to trigger the following notification
“04000”	Start to go to lift.	开始去电梯门口。	info	0.5.2	
“04001”	Succeed to go to lift.	去电梯门口成功了。	info	0.5.2	
“04002”	Failed to go to lift.	去电梯门口失败了。	info	0.5.2	

code	description	Chinese explanation	level	From the following version	Illustration
“04010”	Start to call lift.	开始呼叫电梯。	info	0.5.2	
“04011”	Succeed to call lift.	呼叫电梯成功了。	info	0.5.2	
“04013”	Call lift more than 3 minutes.	呼叫电梯超过 3 分钟了还没来。	warning	0.5.2	
“04020”	Start to take lift.	开始乘坐电梯。	info	0.5.2	
“04021”	Succeed to take lift.	乘坐电梯成功，准备出电梯。	info	0.5.2	
“04023”	Take lift more than 3 minutes.”	乘坐电梯超过 3 分钟了还没到。	warning	0.5.2	
“04030”	Start to enter lift.	开始进电梯。	info	0.5.2	
“04031”	Succeed to enter lift.	进电梯成功。	info	0.5.2	
“04032”	Failed to enter lift.	进电梯失败。	info	0.5.2	
“04033”	There is not enough space in the lift,take the next lift.	电梯里空间不够,等待乘坐下一趟电梯。	info	0.7.3	
“04040”	Start to avoid liftt.	进电梯失败，开始回避电梯。	info	0.5.2	
“04041”	Finish to avoid lift.	回避电梯成功。	info	0.5.2	
“04050”	Start to exit lift.	开始出电梯。	info	0.5.2	
“04051”	Succeed to exit lift.	出电梯成功了。	info	0.5.2	
“04052”	Failed to exit lift.	出电梯失败了。	warning	0.5.2	

code	description	Chinese explanation	level	From the following version	Illustration
“04060”	Start to back to lift.	出电梯失败，开始回到电梯。	info	0.5.2	
“04061”	Succeed to back to lift.	回到电梯成功了。	info	0.5.2	
“04062”	Failed to back to lift.	回到电梯失败了。	warning	0.5.2	
“04070”	Start waiting for the lift to unlock.	开始等待电梯解锁。	info	0.8.1	
“04071”	End waiting for the lift to unlock.	等待电梯解锁结束(成功)。	info	0.8.1	
Status related:					The following notifications do not need to be triggered by mobile tasks
“02000”	Poweroff notice.	将关机断电	“info”	0.5.6	
“02001”	Charge status on.	未充电状态=>充电状态	“info”	0.5.6	
“02002”	Charge status off.	充电状态=>未充电状态	“info”	0.5.6	
“02003”	Estop on.	未急停状态=>急停状态	“info”	0.5.6	
“02004”	Estop off.	急停状态=>未急停状态	“info”	0.5.6	
“02005”	Triggered attitude correction.	姿态校正被触发（此时机器人可能被搬动了）	“warning”	0.7.3	
“02006”	Software shutdown notice.	软件即将关闭	“info”	0.8.2	
“02010”	The robot maybe lost.	机器人可能迷路了	“warning”	0.9.9	There is a possibility of false positives for this notification.
Abnormal state:					The following notifications do not need to be triggered by mobile

code	description	Chinese explanation	level	From the following version	Illustration
					tasks
“03001”	Abnormal object was detected in the robot.	机器人体内检测到异常物体(激光槽内有异物).	“warning”	0.8.2	

Field explanation：

Field	Illustration	Remark
code	Notification ID	As a unique identifier, the code used in the same notification will not change, and new notifications will be expanded in the future
description	English explanation of the notice	Can be used as print information
level	Notification level	Limited to info/warning/error types, the division of each code may be changed in the future
data	Reference information attached to the notice	Software version v0.7.4 new
Chinese explanation	Chinese interpretation of the notice	Only exists in the document, the actual notification will not send this field

Starting from the software version 0.7.4, the notification adds the data field as additional information, see below for details:

code	data	Illustration	From the following version
01001- 01031	“target”:” m”	The name of the current target point	0.7.4
01002/010023/01004	“distance”: 0.2	Task walking distance (m)	0.7.6
01101- 01104	“markers”:[“m1”,” m2”]	Point of the cruise	0.7.4
	“count”:-1	Number of laps to cruise	0.7.4
	“distance_tolerance”:1	Point reach distance tolerance	0.7.4

Example：

```
{
```

```
    "type": "notification",
    "code": "01001",
    "level": "info",
    "description": "The move task is started.",
    "data":{"target":"room_205"}}
```

code	data	Illustration	From the following version
02010	“probability” : 0.7	float type, >0 is the probability of positioning loss	0.9.9

Example:

```
{
    "type": "notification",
    "code": "01001",
    "level": "info",
    "description": "The move task is started.",
    "data":{"target":"room_205"}}
```

11. Set parameters

This interface requires at least software version v0.4.2

Instruction:

```
/api/set_params
```

Parameter:

Name	Illustration	Is it mandatory	Range	Remark	From the following version
max_speed	Maximum travel speed of robot (percentage)	Optional	[0.3, 0.7]	Less than 0.3 to take 0.3, greater than 0.7 to take 0.7	v0.4.2-v0.5.1 (deprecated)
max_speed_ratio	Percentage of robot maximum travel speed	Optional	[0.3, 1.4]	Less than 0.3 to take 0.3, greater than 1.4 to take 1.4	v0.5.2-v0.8.5.1 (deprecated)
max_speed_linear	Maximum linear	Optional	[0.1, 1.0]	Less than 0.1 takes	v0.8.6

Name	Illustration	Is it mandatory	Range	Remark	From the following version
	speed of robot		(m/s)	0.1, greater than 1.0 takes 1.0	
max_speed_angular	Maximum angular velocity of robot	Optional	[0.5, 3.5] (rad/s)	0.5 is less than 0.5, 3.5 is greater than 3.5	v0.8.6

Note: The maximum line/angular speed set by this interface will be combined with the internal parameters of the robot, and the minimum value will take effect in the final operation

The validity period after the successful setting of the parameters of this interface is valid before restarting the software or the whole machine, and it becomes invalid after restarting the software or the whole machine. If you need to take effect permanently, you can manually modify it through the monitoring page configuration management

Interface Description:

Set the parameters of the robot.

It should be noted that after calling this interface, no matter whether it is successful or not, the "status" field in the returned json is "OK". If you want to determine whether the parameter is set successfully, please call interface 12 to get the parameter and check the current value of the parameter after obtaining the parameter.

Example:

Send:

```
/api/set_params?max_speed_linear=0.5
// Set the maximum travel speed of the robot to 0.5 m/s
```

Return:

```
{
    "type": "response",
    "command": "/api/set_params",
    "uuid": "",
    "status": "OK",
    "error_message": ""}
```

12. Get parameters

This interface requires at least software version v0.4.2

Instruction:

```
/api/get_params
```

Parameter:

N/A

Interface Description:

Get the list of parameters that have been set and their current values. For the parameters supported by each version, see interface 11 setting parameters.

Example:

Send:

```
/api/get_params
// Get the parameter list and current value
```

返回:

```
{
   "type": "response",
   "command": "/api/get_params",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {
      "max_speed_linear": 0.5
   }}
```

13.Wireless network interface

13.1Get the list of WiFi currently available to the robot

This interface requires at least software version v0.5.2

Instruction:

```
/api/wifi/list
```

Parameter:

N/A

Interface Description::

Get the list of WiFi currently available to the robot, and return the SSID and signal strength.。

Example:

Send:

```
/api/wifi/list
```

Return:

```
{
   "type": "response",
   "command": "/api/wifi/list",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {
      "SSID1": 50,
      "SSID2": 30,
      "SSID3": 80,
   }}
```

13.2Connect to WiFi

This interface requires at least software version v0.5.2

Instruction:

```
/api/wifi/connect
```

Parameter:

Name	Illustration	Is it mandatory	Range	Remark	From the following version
SSID	WiFi’ s SSID	Required	Currently WiFi		v0.5.2
password	WiFi Password	Optional	SSID corresponding	If you have already connected, you can leave it blank	v0.5.2

Interface Description:

Make the robot connect to the ambient WiFi or switch the currently connected ambient WiFi.

Example:

Send:

```
/api/wifi/connect?SSID=SSID1&password=123456
```

Return:

```
{
    "type": "response",
    "command": "/api/wifi/connect",
    "uuid": "",
    "status": "OK",
    "error_message": "",
}
```

13.3 Get the currently connected WiFi

This interface requires at least software version v0.5.2

Instruction:

```
/api/wifi/get_active_connection
```

Parameter:

N/A

Interface Description:

Get the SSID of the WiFi to which the robot is currently connected.

Example:

Send:

```
/api/wifi/get_active_connection
```

Return:

```
{
   "type": "response",
   "command": "/api/wifi/get_active_connection",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results":"SSID1"   // If you are not currently connected to WiFi, the results
will be empty""}
```

13.4 Obtain the robot IP and wireless network card address

This interface requires at least software version v0.5.2

Instruction:

```
/api/wifi/info
```

Parameter:

N/A

Interface Description:

Obtain the IP address and the physical address of the wireless network card currently allocated by the robot through the environment WiFi.

Example:

Send:

```
/api/wifi/info
```

Return:

```
{
   "type": "response",
   "command": "/api/wifi/info",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results":{
      "IPaddr": "192.168.XXX.XXX", // Empty if WiFi is not connected ""
      "HWaddr": "xx:xx:xx:xx:xx:xx", // Connect the physical address of the WiFi
network card
      },
}
```

13.5 Get detailed information about the wireless list currently available to the robot

This interface requires at least software version v0.7.7

Instruction:

```
/api/wifi/detail_list
```

Parameter:

N/A

Interface Description:

Get the detailed information of the WiFi list currently available to the robot.

Example:

Send:

```
/api/wifi/detail_list
```

Return:

```
{
   "type": "response",
   "command": "/api/wifi/detail_list",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {    // Note: When no available WiFi is detected results={}
      "SSID1": {
         "SSID": "SSID1",  // SSID
         "SIGNAL": 100,  // Signal strength
         "ACTIVE": False, // Connection status True->connected, False->not
connected
         "FREQ": "2462 MHz", // Frequency,  2.4G/5G
         "SECURITY": "WPA2"  // Encryption. Empty means no encryption
       },
      "SSID2": {
         "SSID": "SSID2",
         "SIGNAL": 80,
         "ACTIVE": False,
         "FREQ": "2412 MHz",
         "SECURITY": ""
       },
      "SSID1_5G": {
         "SSID": "SSID1_5G",
         "SIGNAL": 90,
         "ACTIVE": True,
         "FREQ": "5260 MHz",
         "SECURITY": "WPA WPA2"
       }
   }
}
```

14. Map interface

14.1 Get the map list

This interface requires at least software version v0.5.2

Instruction:

```
/api/map/list
```

Parameter:

N/A

Interface Description:

Get all the map names and floors in the robot

Example:

Send:

```
/api/map/list
```

Return:

```
{
   "type": "response",
   "command": "/api/map/list",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results":{
      "map_name_1": [1,2,3,4,5], // There are 1-5 layers in map_name_1
      "map_name_2": [10],
      },
}
```

14.2 Set the current map

This interface requires at least software version v0.5.2

Instruction:

```
/api/map/set_current_map
```

Parameter:

Name	Illustration	Is it Mandatory	Remark	From the following version
hotel_id	Map name	Required		v0.5.2-v0.5.9 (deprecated, replaced by map_name)
map_name	Map name	Required		v0.5.10

Name	Illustration	Is it Mandatory	Remark	From the following version
floor	Floor	Required		v0.5.2

Interface Description:

Set the current map of the robot.

Note: After the setting is successful, the water service will be restarted, so the response may not be received.

Example:

Send:

```
/api/map/set_current_map?hotel_id=map_name_1&floor=5
```

Return:

```
{
    "type": "response",
    "command": "/api/map/set_current_map",
    "uuid": "",
    "status": "OK",
    "error_message": "",
}
```

14.3 Get the current map

This interface requires at least software version v0.5.2

Instruction:

```
/api/map/get_current_map
```

Parameter:

N/A

Interface Description:

Get the current map of the robot.

Example:

Send:

```
/api/map/get_current_map
```

Return:

```
{
   "type": "response",
   "command": "/api/map/get_current_map",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results":{
      "hotel_id": "map_name_1",  // v0.5.2-v0.5.9 (deprecated)
      "map_name": "map_name_1",  // after v0.5.10
      "floor": "5"
      "info": {   // New in v0.7.8, Note: There is no info field when the map fails
to load
         "resolution": 0.05,       // v0.7.8  Resolution (m/pixel)
         "width": 1024,            // v0.7.8 width
         "height": 768,           // v0.7.8 high
         "origin_x": -24.04,       // v0.7.8 Coordinate of lower left corner x
         "origin_y": -12.52       // v0.7.8 Coordinates of the lower left corner
y
      }
   },
}
```

14.4 Get map list details

This interface requires at least software version v0.8.6

Instruction:

```
/api/map/list_info
```

Parameter:

N/A

Interface Description:

Get the detailed information of all the maps in the robot.

Example:

Send:

```
/api/map/list_info
```

Return:

```
{
   "type": "response",
   "command": "/api/map/list_info",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results":{
      "map_name_1": [{"floor": 1, "resolution": 0.02, "height": 1024, "width": 768,
"origin_x": -24.04, "origin_y": -12.52}, {"floor": 2, ...}, ...], // All map
information in map map_name_1
      "map_name_2": [{"floor": 1, ...}, ...], // All map information in map
map_name_2
      ...
      },
}
```

14.5 Given the target point, find the location of the reachable point near the target point

This interface requires at least software version v0.10.7

Instruction:

```
/api/map/accessible_point_query
```

Parameter:

Name	Illustration	Is it mandatory	Remark	From the following version
x	目标点在地图坐标系下的 x 坐标(单位:m)	Required		v0.10.7
y	目标点在地图坐标系下的 y 坐标(单位:m)	Required		v0.10.7

Interface Description:

Given the target point, according to the detection result of the sensor, find the position of the reachable point near the target point.

Note: This interface is only applicable to the current floor of the current map.

Example:

Send:

```
/api/map/accessible_point_query?x=-0.5&y=-0.5
```

Return:

```
{
   "type": "response",
   "command": "/api/map/accessible_point_query",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results":
    {
       "position": {"y": -0.5, "x": -0.5}
    },
}
```

14.6 Given the target point, query the distance to the static map obstacle and the obstacle detected by the sensor

This interface requires at least software version v0.10.7

Instruction:

```
/api/map/distance_probe
```

Parameter:

Name	Illustration	Is it mandatory	Remark	From the following version
x	The x coordinate of the target point in the map coordinate system (unit: m)	Required		v0.10.7

Name	Illustration	Is it mandatory	Remark	From the following version
y	The y coordinate of the target point in the map coordinate system (unit: m)	Required		v0.10.7

Interface Description:

Given a target point, query the distance between the obstacles on the static map and the obstacles detected by the sensor.

In the return result:

Obstacle represents the distance from the target point to the obstacle detected by the sensor, -1: indicates that the target point is too far away to obtain the distance information

static: represents the distance from the target point to the obstacle on the static map.

Example:

Send:

```
/api/map/distance_probe?x=-0.5&y=-0.5
```

Return:

```
{
   "type": "response",
   "command": "/api/map/accessible_point_query",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results":
    {
       "env_dist":
       {
          "obstacle": 0.307566, "static": 0.34
       }
    },
}
```

15.Shut down and restart interface

This interface requires at least software version v0.5.10

Instruction:

```
/api/shutdown
```

Parameter:

Name	Illustration	Is it mandatory	Range	Remark	Software Version
reboot	Whether to restart after shutdown	Optional	true/false	The default is false	v0.5.10
delay	How long does it take to restart after shutdown	Optional	[0, 14400] (unit minute)	It takes effect when reboot is true	v0.8.1

Interface Description:

Call the interface to shut down or restart the robot, a notification will be sent before the power is turned off (see interface 10 Robot Active Notification), and the power will be turned off 10s after the notification is sent. If it is restarting, there will be a 5s interval between power-on and power-off.

Note: the response may not be received

Example:

Send:

```
/api/shutdown   // Shut down
```

Or

```
/api/shutdown?reboot=true      // Restart
```

Send:

```
{
   "type": "response",
   "command": "/api/shutdown",
   "uuid": "",
   "status": "OK",
   "error_message": "",
}
```

16. Software update interface

16.1 Get the current software version

This interface requires at least software version v0.5.5

Instruction:

```
/api/software/get_version
```

Parameter:

N/A

Interface Description:

Get the current software version.

Example:

Send:

```
/api/software/get_version
```

Return on success:

```
{
   "type": "response",
   "command": "/api/software/get_version",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results":"x.x,x",   // current software version
}
```

Return on failure:

```
{
   "type": "response",
   "command": "/api/software/get_version",
   "uuid": "",
   "status": "UNKNOWN_ERROR",
```

```
   "error_message": "xxx",   // Reason for failure
}
```

16.2 Check for updates

This interface requires at least software version v0.5.5

Instruction:

```
/api/software/check_for_update
```

Parameter:

N/A

Interface Description:

Check whether there is a new version that can be updated, return the current version, the latest version and whether it can be updated

Note:

1. The function requires the robot to connect to the network, otherwise it returns to fail.
2. This interface is blocking execution.

Example:

Send:

```
/api/software/check_for_update
```

Return on success:

```
{
   "type": "response",
   "command": "/api/software/check_for_update",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results":{
      "version_latest":"x,x,x",  // current version
      "version_current":"x.x.x", // The latest version
```

```
      "enable_update":bool,  // Can it be updated, true->can be updated,
false->cannot be updated
   },
}
```

Return on failure:

```
{
   "type": "response",
   "command": "/api/software/check_for_update",
   "uuid": "",
   "status": "UNKNOWN_ERROR",
   "error_message": "xxx",    // wrong information
}
```

16.3 Update

This interface requires at least software version v0.5.5

Instruction:

```
/api/software/update
```

Parameter:

N/A

Interface Description:

Update the software to the **latest version**

Note:

1. This function requires the robot to connect to the network, otherwise it will fail.
2. This interface is blocking execution.
3. After the upgrade is successful, the software service will be restarted automatically, and all tcp sockets need to be reconnected.

Example:

Send:

```
/api/software/update
```

Return on success:

```
{
   "type": "response",
   "command": "/api/software/update",
   "uuid": "",
   "status": "OK",
   "error_message": "",
}
```

Return on failure:

```
{
   "type": "response",
   "command": "/api/software/update",
   "uuid": "",
   "status": "UNKNOWN_ERROR",
   "error_message": "xxx",    // wrong information
}
```

16.4 Restart the service

This interface requires at least software version v0.8.2

Instruction:

```
/api/software/restart
```

Parameter:

N/A

Interface Description:

Restart software service

Note:

1. After restarting the software service, all tcp sockets need to be reconnected。

Example:

Send:

```
/api/software/restart
```

Return on success:

```
{
   "type": "response",
   "command": "/api/software/restart",
   "uuid": "",
   "status": "OK",
   "error_message": "",
}
```

17. Set the light strip interface

17.1 Set the brightness of the light strip

This interface can be replaced by the 17.2 setting light strip color interface

This interface requires at least software version v0.6.3

Instruction:

```
/api/LED/set_luminance
```

Parameter:

Name	Illustration	Is it mandatory	Range	Remark	Software version
value	Brightness percentage	Required	[0,100]	Non-uniform change	v0.6.3

Interface Description:

Set the brightness of the LED strip. After setting, it will permanently change the maximum brightness of the strip when it is normal/flashing/breathing.

Note: Although this interface returns success in emergency stop or charging state, the settings may not take effect.

Example:

Send:

```
/api/LED/set_luminance
```

Return on success:

```
{
    "type": "response",
    "command": "/api/LED/set_luminance?value=50",
    "uuid": "",
    "status": "OK",
    "error_message": "",
}
```

Return on failure:

```
{
    "type": "response",
    "command": "/api/LED/set_luminance",
    "uuid": "",
    "status": "UNKNOWN_ERROR",
    "error_message": "xxx",    // Reason for failure
}
```

17.2 Set the color of the light strip

This interface requires at least software version v0.8.7

Instruction:

```
/api/LED/set_color
```

Parameter:

Name	Illustration	Is it mandatory	Range	Remark	Software version
r	Red Value	Required	[0,100]		v0.8.7
g	Green Value	Required	[0,100]		v0.8.7
b	Blue Value	Required	[0,100]		v0.8.7

Interface Description:

Set the color of the LED light strip. After setting, it will permanently change the color of the light strip when it is normal/flashing/breathing. Different colors can be combined according to the value of r/g/b. It should be noted that when rgb is all 0, it cannot take effect. This setting can read and write the flash of the hardware device, and it is not recommended for high frequency use.

Note: Although this interface returns success in emergency stop or charging state, the settings may not take effect.

Example:

Send:

```
/api/LED/set_color?r=0&g=100&b=0  // Set the light strip color to green
```

Return on success:

```
{
   "type": "response",
   "command": "/api/LED/set_color",
   "uuid": "",
   "status": "OK",
   "error_message": "",
}
```

失败返回:

```
{
   "type": "response",
   "command": "/api/LED/set_color",
   "uuid": "",
   "status": "UNKNOWN_ERROR",
   "error_message": "xxx",    // reason for failure
}
```

18. Self-diagnosis interface

18.1 Obtaining self-diagnostic results

This interface requires at least software version v0.6.7

Instruction:

```
/api/diagnosis/get_result
```

Parameter:

N/A

Interface Description:

Get self-diagnostic results。

Example:

Send:

```
/api/diagnosis/get_result
```

Return on success:

```
{
   "type": "response",
   "command": "/api/diagnosis/get_result",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {
      "sensor_core": {  // Diagnosis item: sensor board
         "status": bool,  // The most recent diagnosis result, true->success
false->failure
         "time_stamp": 1511235083.066043,  // Last diagnosis time
         "total_count": 4,  // Total number of diagnoses
       "success_count": 4}, // Number of successful diagnosis
      "motor_core_right": {...},  // Diagnosis item: right motor board
      "motor_core_left": {...},  // Diagnosis item: left motor board
      "radio_core": {...},  // Diagnosis item: wireless board
      "power_core": {...}, // Diagnosis item: power board
      "depth_camera": {...}, // Diagnosis item: Depth camera
      "laser": {...}, // Diagnostic item: laser
      "IMU": {...}, // Diagnosis item: IMU
      "CAN": {...}, // Diagnosis item: CAN module
      "internet": {...}, Diagnosis item: Internet (ping baidu)
      ... // To be added
   }
}
```

Return on failure:

```
{
   "type": "response",
   "command": "/api/diagnosis/get_result",
   "uuid": "",
```

```
   "status": "UNKNOWN_ERROR",
   "error_message": "xxx",    // reason for failure
}
```

19. Get power status interface

This interface requires at least software version v0.7.9

Instruction:

```
/api/get_power_status
```

Parameter:

N/A

Interface Description:

Get battery voltage, charging voltage, current, power and other information

Example:

Send:

```
/api/get_power_status
```

Return:

```
{
   "type": "response",
   "command": "/api/get_power_status",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {
      "battery_capacity": 100, // Battery percentage
      "battery_current": 0.1, // Battery current
      "battery_voltage": 29.5, // Battery voltage
      "charge_voltage": 28.9,  // Charging voltage
      "charger_connected_notice": true, // Is it charging? true->Charging status.
false->uncharged state
      "head_current": 0,         // Power consumption current of host computer
```

```
    }
}

Note:
The battery current symbol is positive for charging, negative for discharging
```

20. Get the robot global path interface

This interface requires at least software version v0.8.2

Instruction:

```
/api/get_planned_path
```

Parameter:

N/A

Interface Description:

Get the global path currently planned by the robot.

Note:

The maximum number of points on the returned path is limited. If the upper limit is exceeded, a certain number of points on the path will be returned on average; if there is no task currently, an empty path will be returned

Example:

Send:

```
/api/get_planned_path
```

Return:

```
{
    "type": "response",
    "command": "/api/get_planned_path",
    "uuid": "",
    "status": "OK",
    "error_message": "",
    "results": {
        "path": [[17.8172,-32.7272], ... [17.8166,-32.7169]]  // Points on the
planned path, up to 2000 points are returned
```

```
}}
```

21. Get elevator status interface

This interface requires at least software version v0.8.5

Instruction:

```
/api/lift_status
```

Parameter:

N/A

接口说明 Interface Description:

Get the status of the elevator the robot is currently riding on

Note:

The call timing of this interface is from "start calling elevator" to "exit elevator success", running_status starts from "wait_lift_outside" and ends with "exit_lift", calls at other times will return a timeout error

Example:

Send:

```
/api/lift_status
```

Return on success:

```
{
   "type": "response",
   "command": "/api/lift_status",
   "uuid": "",
   "status": "OK",
   "error_message": "",
   "results": {
      "current_floor": 1  // The current elevator floor, note: 0 means failure to
get the floor
   }
}
```

22. Get the path interface between two points

This interface requires at least software version v0.9.6

Instruction:

```
/api/make_plan
```

Parameter:

Name	Illustration	Is it mandatory	Range	Remark	From the following version
start_x	The coordinate x of the starting position	Required			
start_y	The coordinate y of the starting position	Required			
start_floor	Starting floor	Required			
goal_x	The coordinate x of the target position	Required			
goal_y	The coordinate y of the target position	Required			
goal_floor	Target location floor	Required			

N/A

Interface Description:

Plan a shortest path from the starting position to the target position, and return the length of the path

Example:

Send:

```
/api/make_plan?start_x=1.0&start_y=1.0&start_floor=1&goal_x=1.0&goal_y=2.0&goal_floor=1
```

Return:


```
{
    "type": "response",
```

```
"command": "/api/make_plan",
"uuid": "",
"status": "OK",
"error_message": "",
"results": {
   "distance": 1.0  // The path length is 1.0 m
}}
```

a. Status code return description (for status of returned data)

Return code	Definition description
OK	Indicates that the response contains valid results
INVALID_REQUEST	Indicates that the provided request is invalid. Common reasons for this status include invalid parameters or parameter values
REQUEST_DENIED	Indicates that your request has been rejected
UNKNOWN_ERROR	Indicates that the request could not be processed due to an error in the server. If you try again, the request may succeed

b. Error message return description (for error_message of returned data)

If the status code is not OK, the response object contains an additional error_message field that explains the reason behind the given status code in more detail.

Note: This field does not guarantee that there will always be content, and its content may change.