REEMAN CALLING API

Multicast

Basic Information

- Multicast IP: 239.0.0.1
- Multicast Port: 7979

• Robot-side Multicast Content

- After the robot enters pairing mode, it multicasts content (JSON):

```
JSON
{
    "hostname":"reeman-test-001",
    "alias":"reeman-test-001",
    "key":"12345678",
    "token":"token",
    "robotType":1:Square low plate(Hussar);2:Moomknight;3:Circular Chassis;4:Flyboat(With QR Code);5:Bigdog;6:Flyboat(Without QR Code);7:Flyboat(Dual laser);8:Bigdog(Dual laser);9:Forklift;
}
```

Parameter	Name Parameter Value	Type Description
hostname	String	Robot ID
encryptKey	String	Data encryption key

Parameter	Name Parameter Value	Type Description
token	String	Generated by the robot during pairing. Must be used in all robot-mobile communications.

MQTT

Basic Information

- Server Address: `mqtt.rmbot.cn`
- Port: 1883

Mobile End Publish

Heartbeat

- Robot wakes up upon receiving heartbeat. If no heartbeat is received for >10s, the robot stops heartbeat and ignores all mobile commands.

Topic: `reeman/calling/phone/{hostname}/v2/heartbeat`

Payload (JSON):

```JSON
{
  "token": "token"
}
```

Request Calling Model Points

Topic: `reeman/calling/phone/{hostname}/v2/points/request/calling_model`

Payload (JSON):

```JSON
{
  "token": "token"
}
```

Request Normal Model Points

Topic: `reeman/calling/phone/{hostname}/v2/points/request/normal_model`

Payload (JSON):

```JSON
{
  "token": "token"
}
```

Request Route Model Points

Topic: `reeman/calling/phone/{hostname}/v2/points/request/route_model`

Payload (JSON):

```JSON
{
  "token": "token"
}
```

Request QR Code Model Points

Topic: `reeman/calling/phone/{hostname}/v2/points/request/qrcode_model`

Payload (JSON):

```
JSON
{
  "token": "token"
}
```

Publish Normal Model Task

Topic: `reeman/calling/phone/{hostname}/v2/task/normal_model`

Payload (JSON):

```
JSON
{
  "token": "token",
  "body": "body"
}
```

Parameter	Name Parameter Value	Type Description
token	String	token: Generated by the robot during pairing.
body	String	AES-encrypted location data, pre-encryption content as follows:

```
JSON
[
    {
        "map":"map1",
        "point":"point1"
    },
    {
        "map":"map2",
        "point":"point2"
    }
]
```

Parameter	Name Parameter Value	Type Description
map	String	Map name: Optional. When the elevator control mode is not enabled, omit this field or pass null.
point	String	Point Name

Publish Route Model Task

Topic: `reeman/calling/phone/{hostname}/v2/task/route_model`

Payload (JSON):

```
JSON
1 {
2     "token":"token",
3     "body":"body"
4 }
```

Parameter	Name Parameter Value	Type Description

token	String	token: generated by the robot during pairing
body	String	AES-encrypted route name (pre-encryption plaintext as follows):

```
JSON
1 "routeName"
```

Publish QR Code Model Task

Topic: `reeman/calling/phone/{hostname}/v2/task/qrcode_model`

Payload (JSON):

```
JSON
1 {
2     "token":"token",
3     "body":"body"
4 }
```

Parameter	Name Parameter Value	Type Description
token	String	token: generated by the robot during pairing
body	String	AES-encrypted location set (pre-encryption): first=pickup point, second=dropoff point, as follows:

```JSON
[
  {
    "first":{
        "map":null,
        "point":"point1"
    },
    "second":{
        "map":null,
        "point":"point2"
    }
    },
    {
     "first":{
    "map":null,
    "point":"point1"
    },
    "second":{
        "map":null,
        "point":"point2"
    }
    ]
```

Parameter	Name Parameter Value	Type Description
	List<android.util.Pair<Point,Point>>	In QR Code mode, location points must be selected in pairs. Each Point object contains two attributes: map and point.
map	String	Map name: Optional. Omit this field or pass null when elevator control mode is disabled.
point	String	point name

Publish Charging Task

Topic: `reeman/calling/phone/{hostname}/v2/task/charge_model`

Payload (JSON):

```
JSON
{
    "token":"token",
    "body":null
}
```

Parameter	Name Parameter Value	Type Description
token	String	token: generated by the robot during pairing
body	String	Request body shall be omitted in recharge mode

Publish Return Task

Topic: `reeman/calling/phone/{hostname}/v2/task/return_model`

Payload (JSON):

```
JSON
{
    "token":"token",
    "body":null
}
```

Parameter	Name Parameter Value	Type Description
token	String	token: generated by the robot during pairing
body	String	No body parameters required

		in [mode] mode

Robot End Publish

Heartbeat

- **Published every 5 seconds after robot wakes up.**

Topic: `reeman/calling/robot/{hostname}/v2/heartbeat`

Payload (JSON):

JSON

```
{
  "hostname": "reeman-001-001",  // Robot ID
  "token": "token",
  "alias": "reeman-test-001",   // Alias
  "level": 99,                  // Battery level (%)
  "lowPower": false,            // Low battery triggered
  "emergencyButton": 0,         // Emergency stop: 0=Pressed, 1=Released
  "chargeState": 1,            // Charging: 1=Not charging, 2=Dock, 3=Cable, 8=Docking, >8=Failed
  "isNavigating": false,       // Navigating?
  "isElevatorMode": false,    // Elevator control enabled? (Reserved)
  "robotType": 4,             // Robot type (see Multicast)
  "liftModelState": 0,         // Lift position: 0=Low, 1=High
  "isLifting": false,          // Lifting? (Forklift: docking pallet)
  "isMapping": false,          // In mapping mode?
  "taskExecuting": false,     // Executing task?
  "currentTask": {            // Current task (null if none)
    "createTime": 1000000000,  // Task creation time (ms)
    "startTime": 1000000000,   // Start time (ms)
    "taskMode": 0,  // 0=Normal, 1=Route, 2=QR Code, 3=Charging, 4=Returning, 5=Calling
    "token": "token",  // Token of task initiator
    "targetPoint": "point1"   // Current target point
  },
  "taskList": [       // Pending tasks
    {
      "createTime": 1000000000,
      "startTime": 1000000000,
      "taskMode": 5,
      "token": "token",
      "targetPoint": "point1"
    },
     ...   ]}
```

Calling Model Points

Topic: `reeman/calling/robot/{hostname}/v2/points/response/calling_model`

Payload (JSON):

```JSON
{
    "token":"token",
    "code":0,
    "body":"body"
}
```

Parameter	Name Parameter Value	Type Description
token	String	token: generated by the robot during pairing
code	Integer	tatus Codes: 0: Success - Decrypt body using secret key to access location data 1001-1999: Error - Body contains unencrypted error message (no decryption required)
body	String	AES-encrypted location data (decrypted payload shown below):

```JSON
{
    "elevatorModeSwitch":true,
    "model":{
        "map1":[
                "point1",
                "point2",
                ]
    }
}
```

Parameter	Name Parameter Value	Type Description
elevatorModeSwitch	Boolean	Elevator control switch: When enabled, display location points according to the specified map name.
model	Map<String,List<String>>	"key: map_name, value: list of waypoints

Normal Model Points

Topic: `reeman/calling/robot/{hostname}/v2/points/response/normal_model`

Payload (JSON):

```json
{
    "token":"token",
    "code":0,
    "body":"body"
}
```

Parameter	Name Parameter Value	Type Description
token	String	token: generated by the robot during pairing
code	Integer	Status Codes: 0: Success - Decrypt body payload using secret key to access location data 1001-1999: Error - Body contains plaintext error message (skip decryption)
body	String	Decrypted location data (after AES decryption):

```JSON
{
  "elevatorModeSwitch":true,
  "model":{
        "map1":[
              "point1",
              "point2"
              ],
        "map2":[
              "point1",
              "point2"
              ]
        }
}
```

Parameter	Name Parameter Value	Type Description
elevatorModeSwitch	Boolean	Elevator control switch: When activated, display location markers according to the designated map name.
model	Map<String,List<String>	key: map_name, value: array of waypoints

Route Model Routes

Topic: `reeman/calling/robot/{hostname}/v2/points/response/route_model`

Payload (JSON):

```JSON
{
    "token":"token",
    "code":0,
    "body":"body"
}
```

Parameter	Name Parameter Value	Type Description
token	String	token: generated by the robot during pairing
code	Integer	Status Codes: 0: Success - Decrypt body using secret key to access location data 1001-1999: Error - Body contains plaintext error message (skip decryption)
body	String	AES-decrypted location data (plaintext output):

```JSON
["route1","route2"]
```

Parameter	Name Parameter Value	Type Description
	List<String>	Route Name List

QR Code Model Points

Topic: `reeman/calling/robot/{hostname}/v2/points/response/qrcode_model`

Payload (JSON):

```JSON
{
    "token":"token",
    "code":0,
    "body":"body"
    }
```

Parameter	Name Parameter Value	Type Description
token	String	token: generated by the robot during pairing
code	Integer	Status Codes: 200 OK: Success - Decrypt body using secret key to retrieve location data 4xx/5xx: Error - Body contains plaintext error message (no decryption needed)
body	String	AES-decrypted location data (QR Code mode currently doesn't support elevator control. If added later, the body structure remains unchanged):

JSON

```
{
    "elevatorModeSwitch":true,
    "model":{
        "map1":[
            "point1",
            "point2"
        ]
    }
}
```

Parameter	Name Parameter Value	Type Description
elevatorModeSwitch	Boolean	Elevator Control Switch: When enabled, displays location-specific waypoints according to the designated map name.

model	Map<String,List<String>>	key: map_name, value: waypoint_list

Task Execution Response

Topic: `reeman/calling/robot/{hostname}/v2/task/response`

Payload (JSON):

```JSON
{
    "token":"token",
    "code":0,
    "body":"body"
}
```

Parameter	Name Parameter Value	Type Description
token	String	token: robot-generated during pairing
code	Integer	Status Codes: 0: Execution started 2001-2999: Execution failed to initiate
body	String	AES-decrypted task result (String type): e.g. "Task started" / "Emergency stop pressed - task aborted"

Status Codes

Code	Description
0	Success

Code	Description
1001	Failed to get points (data source issue)
1002	Failed to get points (elevator control error)
2001	Task failed (data source issue)
2002	Task failed (data source issue)
2003	Task failed (invalid state: emergency stop, low battery, busy, lift not reset, etc.)