M4 HTTP 和 WebSocket 接口文档

M4 HTTP 和 WebSocket 接口文档

1.1 概述

M4 对外提供多种协议的接口。本文档主要讨论 HTTP 和 WebSocket 接口。

1.1.1 HTTP 风格概述

M4 的 HTTP 接口多数的请求和响应正文是 JSON （application/json）。并且根据 HTTP 规范，一定是 UTF-8 编码的（即 application/json; charset=utf-8）。 HTTP 接口参考了 Restful 风格。但考虑到国内的接受程度，未能全面采用 Restful 风格。比如还是尽量使用了 GET 和 POST 请求，少用 PUT/DELETE 请求。

但在 HTTP 响应码上，一般：

200 表示请求成功。

201 现在已尽量避免使用 201。规范含义是创建成功。但现在尽量都尽量改用 200。

400 表示请求错误（Bad Request）。

404 表示请求的 URL 不存在。

401 表示身份验证失败（未登录）。

403 表示已登录但权限不够。

500 表示服务器内部错误，即服务器出 BUG 了。

502 表示服务不可用。

504 表示请求超时。

对于 400，即请求错误。响应正文是一个 JSON。有 code 和 message 两个字段，给出错误码和错误消息。code 可能为空，但 message 一定有内容。 { "code": "NoSuchRobot", "message": "找不到机器人 '2032'" }

1.1.2 时间

为了方便各语音解析，接口返回的日期时间字段，一般是一个 Unix timestamp 长整数，毫秒精度，如 1699200000000。注意需要一个 8 字节整数才能表示。 请求中的日期时间，我们会“尽力解析”，包括 ISO 8601 等。目前支持以下格式： "yyyy-MM-dd'T'HH:mm:ss.SSSXXX", "yyyy-MM-dd'T'HH:mm:ss.SSSZ", "yyyy-MM-dd'T'HH:mm:ssX", "yyyy-MM-dd HH:mm:ss", "yyyy-MM-dd HH:mm:ss.SSS", "yyyy-MM-dd HH:mm", "yyyy-MM-dd", "yyyy/MM/dd",

"HH:mm:ss"

1.2 安全相关接口

1.2.1 PING 接口

GET /api/ping

用来检查是否能与服务器通讯并且处于已登录状态。如果未登录（或过期）返回 401。如果已登录，返回用户 ID、用户名等信息。如

{

"id": "**admin**", "username": "admin", "roAdmin": true,

"permissions": null

}

1.2.2 登录接口

POST /api/sign-in

请求正文，携带用户名和密码，如：

{

```
  "username": "admin",
```

"password": "admin"

}

登录成功响应 200，登录失败响应 400，给出错误原因。 登录成功响应的正文中，给出用户 ID，例如,

{

```
"userId": "__admin__",
"userToken": "n6F0HgxprbgTvH1AJ95qmQ5H"
```

}

1.2.3 应用安全验证

通过用户名密码登录、获取 token/cookie 适合人类用户使用。对于程序构成的客户端，可以在每次发送请求时携带身份信息，不需要先登录。 身份信息放在 HTTP 请求头上。xyy-app-id 给出应用 ID，xyy-app-key 给出应用的秘钥。 获取应用 ID 和 密钥的方式步骤如下图，管理员在 M4 的 代理用户 菜单中添加对应的应用 ID 和密钥。 QvW1bLyiWojNaMxwTq4cFVpVnOd.png

在请求 M4 的 API 接口中，将其放在请求头中即可，如图： MIVNbq3F5oczrMxLS3FcA6TQndd.png

1.3 业务对象增删改查

M4 系统整体建构在一个低代码系统上。系统内的各种概念，比如用户、库位、机器人、任务、订单等，称为一个业务对象。在国内外软件领域，也称为实体（Entity）或领域对象（Domain Object）。 我们为业务对象提供一批统一的接口，只要接入这套接口，几乎可以查询和修改系统内的所有数据。

1.3.1 查询条件

M4 接口支持构造任意你想表达的查询条件。可以用于分页查询、批量查询，也可以用于更新、删除条件。例如，下面会将批量查询，如果要查询供应商为 QA0001 且单据状态为 Created 或 Pending 的单据:

{

```
    "type": "Compound",
```

"items": [

{

```
            "type": "General",
            "field1": "vendor",
            "operator": "Eq",
```

"value": "QA0001"

```
        },
```

{

```
            "type": "General",
            "field1": "status",
            "operator": "In",
```

"value": [

```
                "Created",
```

"Pending"

]

}

]

}

查询条件类似 SQL 查询。有三种类型：全部、复合、通用。

全部，即查询全部数据：

{ "type": "All" }

复合，可以构造 AND 或 OR 条件。or 为 false 即是 AND，or 为 true 即是 OR。or 默认是 false，即默认是 AND。 { "type": "Compound", "or": false, "items": [...] } 通用，可以将指定字段与特定条件匹配。

{

```
"type": "General",
"field1": "vendor",
"operator": "Eq",
```

"value": "QA0001"

}

field1 即字段名。operator 是比较条件。value 是比较值。

operator 类似 SQL 运算符，可以使用：

Eq 相等

Ne 不等于

Gt 大于

Gte 大于等于

Lt 小于

Lte 小于等于

In 包含多个值，value 传一个数组。 Between 用于数字和日期类型，value 传一个数组，两个值，闭闭区间。

Contain 包含，用于文本，包含给定文本

ContainIgnoreCase 忽略大小写

Start 用于文本，匹配文本开头

End 用于文本，匹配文本结束

Null 值精确等于 null

NotNull 值精确不等于 null

Empty 用于文本，字段为 null 或是空POST /api/entity/create/many字符串

NotEmpty 上面的反转

CurrentUser 匹配当前用户 ID，不需要传 value CurrentUsername 匹配当前用户的用户名，不需要传 value

ThisWeek 筛选指定日期为本周内

一些列子：

查询我创建的单据：

{

```
"type": "General",
"field1": "createdBy",
```

"operator": "CurrentUser"

}

查询本周创建的单据：

{

```
"type": "General",
"field1": "createdOn",
```

"operator": "ThisWeek"

}

查询 2023-01-01 到 2023-01-31 创建的单据：

{

```
"type": "General",
"field1": "createdOn",
"operator": "Between",
"value": [ "2023-01-01", "2023-01-31" ]
```

}

1.3.2 分页查询

分页查询，顾名思义，就是分页查询。可以指定查询条件、排序方式、返回部分字段。

POST /api/entity/find/page

注意，为了携带复杂查询，此接口是 POST 请求不是 GET 请求。

请求正文示例：

{

```
"entityName": "HumanUser",
"query": null,
"pageNo": 1,
"pageSize": 50,
"projection": null,
```

"sort": [

"-id"

]

}

entityName 是业务对象的名字。 query 是查询条件。null 表示查询全部。 pageNo 是页码，从 1 开始。 pageSize 是一页几条数据。 projection 可以裁剪业务对象字段。null 表示返回全部字段。 sort 是排序方式。 如果想只返回几个字段，可以通过 projection 设置：

"projection": ["username", "age"] sort 支持按多个字段排序。字段名前 - 表示倒序。+ 或空表示正序。例如，先按创建创建倒序排序，再按 id 正序排序。

"sort": ["-createdOn", "+id"]

响应示例：

{

```
"pageNo": 1,
"pageSize": 50,
"total": 200,
```

"page": [

{

```
        "username": "test1",
        "email": null,
        "createdBy": "__admin__",
        "createdOn": 1699804800000,
        "id": "5DB8C406735B40C4A81CB8FCFF8656E8"
```

}

]

}

响应里，total 是一共有多条数据。page 是当前页数组。每个元素表示业务对象，这里为了文档长度，删除了其他业务对象，只保留了一个。

1.3.3 查询数量

返回符合条件的业务对象数量。

POST /api/entity/find/count

注意，为了携带复杂查询，此接口是 POST 请求不是 GET 请求。

请求正文示例：

{

```
"entityName": "HumanUser",
```

"query": null

}

entityName 是业务对象的名字。 query 是查询条件。null 表示查询全部。

响应示例：

{"count":10}

1.3.4 批量查询

批量返回符合条件的多个业务对象。可以限制最多返回几个。与分页一样，支持排序、筛选字段。

POST /api/entity/find/many

注意，为了携带复杂查询，此接口是 POST 请求不是 GET 请求。

请求正文示例：

{

```
"entityName": "HumanUser",
"query": null,
"limit": 10,
"projection": null,
```

"sort": [

"-id"

]

}

entityName 是业务对象的名字。 query 是查询条件。null 表示查询全部。 limit 是最多返回多少个对象。null 或负数表示不限制。 skip 表示跳过前面的 N 个对象。null 忽略。 projection 可以裁剪业务对象字段。null 表示返回全部字段。 sort 是排序方式。

响应示例：

[

{

```
    "username": "test1",
    "email": null,
    "createdBy": "__admin__",
    "createdOn": 1699804800000,
    "id": "5DB8C406735B40C4A81CB8FCFF8656E8"
```

}

]

响应是一个数组。

1.3.5 单个查询

单个查询有两种形式，一种是指定 id，一种是指定查询条件。

POST /api/entity/find/one

注意，为了携带复杂查询，此接口是 POST 请求不是 GET 请求。

请求正文示例：

指定 ID 查询：

{

```
"entityName": "HumanUser",
"id": "OKLD2022-01-02-01221",
"projection": null,
```

"sort": [

"-id"

]

}

也可以指定查询条件，跟批量查询一样，传 query / skip 字段。用于实现一些边界问题，如"查询符合条件的第 4 个对象"。

响应正文：

如果有指定的一个对象，返回：

{

"entityValue": {

```
    "username": "test1",
    "email": null,
    "createdBy": "__admin__",
    "createdOn": 1699804800000,
    "id": "5DB8C406735B40C4A81CB8FCFF8656E8"
```

}

}

如果找不到，返回：

{

"entityValue": null

}

1.3.6 创建单个

POST /api/entity/create/one

请求正文实例：

{

```
"entityName": "HumanUser",
```

"entityValue": {

```
    "username": "test2",
```

"password": "12345678"

}

}

entityName 是业务对象的名字。entityValue 是要新增的业务对象本身。

响应是新增业务对象的 ID：

{"id":"3524032D91A04700911F55E00BA03294"}

1.3.7 创建多个

POST /api/entity/create/many

请求正文实例：

{

```
"entityName": "HumanUser",
```

"entityValues": [

{

```
        "username": "test2",
```

"password": "12345678"

```
    },
```

{

```
        "username": "test3",
```

"password": "12345678"

}

]

}

entityName 是业务对象的名字。entityValues 是要新增的业务对象数组。

响应是新增业务对象的 ID 列表：

{

"ids": [

```
    "3524032D91A04700911F55E00BA03294",
    "3524032D91A04700911F55E00BA03295"
```

]

}

1.3.8 更新单个

POST /api/entity/update/one

请求正文实例：

{

```
"entityName": "HumanUser",
"id": "3524032D91A04700911F55E00BA03294",
```

"update": {

```
    "roAdmin": false,
    "btDisabled": false,
```

}

}

entityName 是业务对象的名字。id 是更新对象的 ID。update 是业务对象需要更新的部分字段（对，可以只传需要更新的字段）。

响应是成功更新的对象数量：

{"updatedCount":1}

1.3.9 更新多个

POST /api/entity/update/many

请求正文实例：

{

```
"entityName": "HumanUser",
```

"query": {

```
    "type": "General",
    "field1": "id",
    "operator": "In",
```

"value": [

```
        "5DB8C406735B40C4A81CB8FCFF8656E8",
        "3524032D91A04700911F55E00BA03294"
```

]

```
},
```

"update": {

"disabled": true

}

}

entityName 是业务对象的名字。query 是筛选条件。update 是业务对象需要更新的部分字段。此外还可以传 limit 字段，限制最多更新几个业务对象。

响应是成功更新的对象数量：

{"updatedCount":2}

1.3.10 删除单个

POST /api/entity/remove/one

请求正文实例：

{

```
"entityName": "HumanUser",
"id": "3524032D91A04700911F55E00BA03294"
```

}

entityName 是业务对象的名字。id 是更新对象的 ID。

响应是实际删除的对象数量：

{"removedCount":1}

1.3.11 删除多个

POST /api/entity/remove/many

请求正文实例：

{

```
"entityName": "HumanUser",
```

"query": {

```
    "type": "General",
    "field1": "id",
    "operator": "In",
```

"value": [

```
        "5DB8C406735B40C4A81CB8FCFF8656E8",
        "3524032D91A04700911F55E00BA03294"
```

]

}

}

entityName 是业务对象的名字。query 是筛选条件。此外还可以传 limit 字段，限制最多删除几个业务对象。

响应是实际删除的对象数量：

{"removedCount":2}

1.3.12 导出 Excel

POST /api/entity/export

请求正文实例：

{

```
"entityName": "HumanUser",
```

"fields": [

```
    "id",
    "username",
    "email",
```

"truename"

```
],
```

"query": null

}

entityName 是业务对象的名字。query 是筛选条件。fields 是要导出的字段/列。

响应示例：

{"path":"tmp/export1282539A910D42E88AC19FB3B52B8643.xlsx"} path 是文件下载路径。可以用文件下载接口获取。

1.3.13 组合查询

同时以不同的查询条件，查不同的实体，比如同时查询单个、查询多个、分页查询。

POST /api/entity/find/batch

请求正文示例：

[

{

```
    "type": "One",
```

"req": {

```
        "entityName": "FbBin",
        "query": {...},
```

"sort": [

"-id"

]

}

```
},
```

{

```
    "type": "Many",
```

"req": {

```
        "entityName": "FbBin",
        "query": {...},
```

"sort": [

"-id"

]

}

```
},
```

{

```
    "type": "Page",
```

"req": {

```
        "entityName": "FbBin",
        "query": {...},
        "pageNo": 1,
        "pageSize": 10,
```

"sort": [

"-id"

]

}

}

]

请求正文是 JSON 数组，由多个 JSON 对象组成，每个 JSON 对象的字段有： type 当前 JSON 对象查询的查询模式，有以下几种枚举

One 查询单个

Many 批量查询

Page 分页查询

req 需要与 type 匹配，请求格式参考 单个查询、批量查询、分页查询 的请求报文内容。

响应示例：

[

// One 的查询结果

```
{...},
```

// Many 的查询结果

[

```
    {...},
    {...},
```

{...}

```
],
```

// Page 的查询结果

{

```
    "pageNo": 1,
    "pageSize": 10,
    "total": 138,
```

"page":[

```
            {...},
            {...},
```

{...}

]

}

]

响应内容是 JSON 数组，响应数组中的顺序与请求数组中的顺序相同。

1.4 文件上传下载

1.4.1 文件上传

POST /api/files/upload

上传文件采用的格式是 multipart/form-data。文件放在 f0 字段里。一次只能上传一个文件。

响应示例：

{

```
"originalName": "test.png",
"size": 7182,
"path": "upload/202311318/5DAED8FC3BEC498BA42CA36282E1DC0E.png"
```

}

originalName 是文件原始名称。size 是文件大小，单位字节。path 是后续获取此文件的路径。

1.4.2 文件下载/读取

GET /api/files/get/

path 是之前上传时返回的路径。

1.5 机器人管理相关（二代）

1.5.1 获取所有机器人的所有信息

GET /api/robot/rachel/all-all

响应是一个数组。每个元素表示一个机器人：

{

"id": "AMB-01", // 机器人 ID

```
"systemConfig": {}, // 参见“移动机器人系统配置”业务对象
"runtimeRecord": {}, // 机器人任务状态
"selfReport": { // 机器人自身报告，可能为 null，表示没收到任何报文
  "error": false, // true 表示与机器人连接故障
```

"errorMsg": "", // 故障详情

"main": {}, // 主要信息

"rawReport": {}, // 机器人报告原始报文

```
  "timestamp": 1699200000000 // 收到报告的时间
},
```

}

systemConfig 是机器人系统配置信息。如厂商、停用、不接单、最大载货数、类别、标签等。参见“移动机器人系统配置”业务对象。 runtimeRecord 是机器人任务状态。如机器人身上载货/库位情况、调度执行状态、当前运单等。参见“移动机器人运行记录”业务对象。 main 是机器人报告的主要部分。对所有厂商的机器人做了统一化处理。主要包括：

battery 电量，从 0 到 1

x 当前位置 x

y 当前位置 y

direction 当前机器人方向

currentSite 当所在站点/点位（逻辑点位名）

blocked 被阻挡

charging 充电中

selfReport 报警信息

level 报警等级

code 报警码编号

message 报警内容

times 出现次数

timestamp 报警时间

1.5.2 向系统中添加机器人

POST /api/robot/rachel/add-robots 请求正文是一个数组。每个元素表示一个机器人配置。字段参见“移动机器人系统配置”业务对象。

1.5.3 从系统中删除机器人

POST /api/robot/rachel/remove-robots 请求正文是一个数组。每个元素是要删除的机器人名（即 ID）。

1.5.4 更新单个机器人配置

POST /api/robot/rachel/update-robot

请求正文：

{

"id": "AMB-01", // 机器人 ID

```
"systemConfig": { ... }, // 参见“移动机器人系统配置”业务对象
```

}

例如，如果要禁用机器人，只需要：

{

```
"id": "AMB-01",
```

"systemConfig": {

"disabled": true

```
},
```

}

1.6 Modbus HTTP 接口

其他系统调用 M4 的 API 接口的方式，来控制基于 Modbus 协议的外部设备，此类的接口验证可以在 M4 注册 APP ID 来免登录，注册方式请见 1.2.3应用安全验证。
DzUqbzQmvoLwYfx0A1TcZMsynJf.png

1.6.1 读 Modbus

POST /api/wcs/plc/modbus/read

请求示例：查询 PLC 中地址位从 35 开始查询 4 位的数值，失败重试 1 次，间隔 1s。

{

```
"deviceName": "PLC",
"maxRetry": 1,
"retryDelay": 1000,
```

"commands": [{

```
    "code": 3,
    "address": 35,
    "qty": 4,
```

"slaveId": 0

}]

}

}

返回示例：地址位 35、36、37、38 的参数分别为 1、111111、0、0。

[

[1,11111,0,0]

]

1.6.2 写 Modbus

POST /api/wcs/plc/modbus/write 请求示例：PLC 中地址位为 35 写入 1。

{

```
"deviceName": "PLC",
"maxRetry": 1,
"retryDelay": 1000,
```

"commands": [{

```
    "code": 6,
    "address": 35,
    "slaveId": 0,
```

"values":[1]

}]

}

1.7 统计

1.7.1 查询周期报表（时间序列数值报表）

HTTP

POST /api/stats/timeline

请求示例：

{

```
"subjects": ["RobotChargeTimes", "RobotChargeDuration"],
"targets": ["AMB-01"],
```

"periodType": "Hour", // 统计周期

```
"timepointStart": "2024-01-01 12H", // 时间点 - 起始
"timepointEnd": "2024-01-01 13H" // 时间点 - 结束
```

}

subjects 是统计科目列表。数组，可以指定多个科目，详见 周期报表支持的科目 。该参数为空查询所有科目，注意可能消耗系统大量资源！

targets 是统计对象列表。例如，机器人名称、某个模块的名称。数组，可以指定多个对象。该参数未空查询所有对象，注意可能消耗系统大量资源！

periodType 是统计周期类型，必填，填一个。可选的周期类型有： "Hour"，时报。 "Day"，日报。 "Week"，周报。 "Month"，月报。 "Quarter"，季报。 "Year"，年报。

接下来还可以指定查询的时间范围。有两种指定方式，通过时间点或时间戳。

时间点表示特定统计周围类型、指定周期的时间表示，包括：

时报时间点："2024-01-01 12H"，2024 年 1 月 1 号，12:00 到 13:00。 日报时间点："2024-01-01"，2024 年 1 月 1 号。 周报时间点："2024 W1"，2024 年第 1 周。W 后是某年底 N 周。 月报时间点："2024-01"，2024 年 1 月。 季报时间点："2024 Q1"，2024 年第 1 季度。 年报时间点："2024"，2024 年。

可以通过 timepointStart 和 timepointEnd 指定开始和结束的时间点（含起始和结束时间，即是 <= 和 >=，不是 < 和 >。）可以只指定开始或结束，或都不指定。都不指定即查询所有，注意可能消耗系统大量资源。

通过时间点查询的请求示例：

// 查询系统中记录的 “AMB-01” 的所有 “充电次数” 和 “里程” 的时报数据。

{

```
"subjects": [   ],
"targets": ["AMB-01"],
```

"periodType": "Hour"

}

// 通过 “时间点”，查询 2024 年 1 月 1 日 12 点及以后的时报数据。

{

```
"subjects": ["RobotChargeTimes", "RobotChargeDuration"],
"targets": ["AMB-01"],
"periodType": "Hour",
"timestampStart": "2024-01-01 12H"
```

}

// 通过 “时间点”，查询 2024 年 1 月 1 日 12 点及以前的时报数据。

{

```
"subjects": ["RobotChargeTimes", "RobotChargeDuration"],
"targets": ["AMB-01"],
"periodType": "Hour",
"timestampEnd": "2024-01-01 12H"
```

}

// 通过 “时间点”，查询 2024 年 1 月 1 日 12 点到 19 点之间的时报数据。

{

```
"subjects": ["RobotChargeTimes", "RobotChargeDuration"],
"targets": ["AMB-01"],
"periodType": "Hour",
"timestampStart": "2024-01-01 12H",
"timestampEnd": "2024-01-01 19H"
```

}

除了通过时间点，还可以通过时间戳指定统计的时间范围。时间戳即传一个时刻，到年月日的精度，或到秒的精度。可以使用常见合法的日期格式，包括 UNIX 时间戳。时间戳范围通过 timestampStart 和 timestampEnd 指定。可以只指定开头或结束，或都不指定。示例： // 查询系统中记录的 “AMB-01” 的所有 “充电次数” 和 “里程” 的时报数据。

{

```
"subjects": ["RobotChargeTimes", "RobotChargeDuration"],
"targets": ["AMB-01"],
```

"periodType": "Hour"

}

// 通过 “时间戳”，查询 “2024-10-11T08:00:00.000z” 及以后的时报数据。

{

```
"subjects": ["RobotChargeTimes", "RobotChargeDuration"],
"targets": ["AMB-01"],
"periodType": "Hour",
"timestampStart": "2024-10-11T08:00:00.000z"
```

}

// 通过 “时间戳”，查询 “2024-10-11T08:00:00.000z” 及以前的时报数据。

{

```
"subjects": ["RobotChargeTimes", "RobotChargeDuration"],
"targets": ["AMB-01"],
"periodType": "Hour",
"timestampEnd": "2024-10-21T08:00:00.000z"
```

}

// 通过 “时间戳”，查询 “2024-10-11T08:00:00.000z” 到 “2024-10-15T08:00:00.000z” 之间的时报数据。

{

```
"subjects": ["RobotChargeTimes", "RobotChargeDuration"],
"targets": ["AMB-01"],
"periodType": "Hour",
"timestampStart": "2024-10-11T08:00:00.000z",
"timestampEnd": "2024-10-15T08:00:00.000z"
```

}

响应示例：

[

{

```
    "id": "67171585C82A850DB9B8FF45",
    "value": "2",
    "subject": "RobotChargeTimes",
    "startedOn": 1729472400000,
    "finishedOn": 1729475999999,
    "createdBy": "",
    "modifiedBy": "",
    "createdOn": 1729566085180,
    "modifiedOn": 1729566085180,
    "version": 0,
    "target": "AMB-01",
    "periodType": "Hour",
```

"period": "2024-10-21 09"

```
},
```

{

```
    "id": "67171585C82A850DB9B8FF49",
    "value": "15159",
    "subject": "RobotChargeDuration",
    "startedOn": 1729472400000,
    "finishedOn": 1729475999999,
    "createdBy": "",
    "modifiedBy": "",
    "createdOn": 1729566085206,
    "modifiedOn": 1729566085206,
    "version": 0,
    "target": "AMB-01",
    "periodType": "Hour",
```


"period": "2024-10-21 09"

}

]

target 是统计对象的名称，与请求正文中 targets 的值相关。 subject 是统计科目，与请求正文中 subjects 的值相关。 periodType 是统计周期类型，与请求正文中 periodType 的值相同。 period 是统计的时间点，与请求正文中 timepoint 的值相关。 value 是当前统计记录对应的值。

WebSocket

ws://host:port/wsm

请求示例：

{

```
"action": "Robot::StatsTimelineValueReport::Query",
"content": "序列化之后的查询参数" // 参考 HTTP 处的请求示例。
```

}

序列化之后的 content 可参考 HTTP 处的 请求示例 。

响应示例：

{

```
"action":"Robot::StatsTimelineValueReport::Reply",
"content": "序列化之后的报表数据" // 参考 HTTP 处的响应示例。
```

}

序列化之后的 content 可参考 HTTP 处的 响应示例 。

1.7.2 周期报表支持的科目

类别

科目编码

名称

说明

机器人

RobotChargeTimes

充电次数

机器人

RobotChargeDuration

充电时长

单位：小时。

机器人

RobotMileage

新增里程

单位：米。

机器人

RobotWorkTimes

工作次数

每创建一个路径导航任务算工作一次。

机器人

RobotWorkDuration

工作时长

单位：小时。

机器人

空闲时长

单位：小时。

机器人

RobotErrorTimes

故障次数

机器人

RobotErrorDuration

故障时长

单位：小时。

机器人

RobotEmptyDuration

空载时长

机器人未载货的时间。单位：小时。

机器人

RobotLoadDuration

载货时长

机器人有载货（不管载几个）的时间。单位：小时。

机器人

RobotOnlineDuration

在线时长

与 M4 正常连接、上报状态视为在线。单位：小时。

机器人

RobotIdleRate

空闲率

空闲率 = 空闲时长 / 在线时长

机器人

RobotNoLoadRate

空载率

空载率 = 空载时长 / 在线时长

机器人

RobotFailureRate

故障率

故障率 = 故障时长 / 在线时长

猎鹰任务

FalconTaskCreate

生成的猎鹰任务记录数量

猎鹰任务

FalconTaskDone

完成的猎鹰任务记录数量

猎鹰任务

FalconTaskError

异常的猎鹰任务记录数量

包括被取消和终止的猎鹰任务记录的总数。

WMS

QsInboundOrderCount

入库单数

单据数

WMS

QsInboundInvQty

入库数量

所有物料的入库数量合计

WMS

QsOutboundOrderCount

出库单数

单据数

WMS

QsOutboundInvQty

出库数量

所有物料的出库数量合计

WMS

QsStatsMaterialCount

期末库存物料种数

不区分物料

WMS

QsStatsInvSummary

期末库存总数量

不区分物料

附录 A：如何获取业务对象详情

通过界面，打开菜单《业务对象首页》，在列表中找到想要了解的业务对象。
OQvfbh53EozMh7xPqEZcziZCnzh.png

可以了解业务对象基本详情、字段等信息。 VjcVbwaclorIxexXKr0c2kFInWg.png