

二次开发接口说明 V0.4.3

开发者可通过标准的ROS2话题（Topic）、服务（Service）和动作（Action）机制，便捷地获取机器人状态、发送控制指令，实现复杂任务的编程与集成。

Topic：订阅式话题，接收方订阅某个消息，发送方根据订阅列表向接收方发送消息，主要用于中高频或持续的数据交互。

Service：问答式服务，通过请求实现数据获取或操作。用于低频或功能切换时的数据交互。

Action：持续式动作，用于长时间运行的任务，支持任务执行过程中的状态反馈和取消操作，适用于复杂动作控制和任务执行。

接口说明

1. 底层关节控制接口

底层关节控制接口用于实时获取全身关节状态（位置/速度/力矩）并提供多模式控制（位置/力矩/力位混合）。

消息格式遵循 ROS2 系统里 sensor_msgs 功能包中定义的 JointState 消息类型的规范。

真机

Topic	/feedback/joint
生效版本	≥V0.2
接入方式	订阅（subscribe）
接口说明	实时获取每个关节（包括臂部、腿部、腰部、颈部等）的位置（单位为弧度）、速度（单位为弧度/秒）和力矩（单位为牛米）信息。
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息
	string[] name：获取当前状态的关节名称，格式为[关节1名称，关节2名称，...，关节20名称]
	float64[] position：每个关节当前的位置信息，反映每个关节的旋转角度，格式为[关节1位置，关节2位置，...，关节20位置]，单位是弧度，取值范围是 $[-\pi, \pi]$
	float64[] velocity：每个关节当前的运动速度，格式为[关节1速度，关节2速度，...，关节20速度]，单位是弧度/秒，取值范围是 $[-\pi, \pi]$

	float64[] effort：每个关节当前所承受的力矩，反映关节电机的输出力，格式为[关节1力矩，关节2力矩，...，关节20力矩]，单位是牛米，取值范围是[-100.0, 100.0]
--	---

Topic	/control/joint_position
生效版本	≥V0.2
接入方式	发布（publish） 消息类型：joint_interfaces/msg/JointPositionControl
接口说明	精确控制各关节到达目标角度
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息 string[] name：控制的关节名称，格式为[关节1名称，关节2名称，...，关节20名称] float64[] position：控制各关节运动至目标角度，格式为[关节1位置，关节2位置，...，关节20位置]，单位是弧度（rad），取值范围是[-π, π]

仿真环境

Topic	/feedback/joint_sim
生效版本	≥V0.2
接入方式	订阅（subscribe）
接口说明	实时获取每个关节（包括臂部、腿部、腰部、颈部等）的位置（单位为弧度）、速度（单位为弧度/秒）和力矩（单位为牛米）信息。
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息 string[] name：获取当前状态的关节名称，格式为[关节1名称，关节2名称，...，关节20名称] float64[] position：每个关节当前的位置信息，反映每个关节的旋转角度，格式为[关节1位置，关节2位置，...，关节20位置]，单位是弧度，取值范围是[-π, π] float64[] velocity：每个关节当前的运动速度，格式为[关节1速度，关节2速度，...，关节20速度]，单位是弧度/秒，取值范围是[-π, π] float64[] effort：每个关节当前所承受的力矩，反映关节电机的输出力，格式为[关节1力矩，关节2力矩，...，关节20力矩]，单位是牛米，取值范围是[-100.0, 100.0]

Topic	/control/joint_position_sim
生效版本	≥V0.2
接入方式	发布（publish）
接口说明	精确控制各关节到达目标角度
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息
	string[] name：控制的关节名称，格式为[关节1名称，关节2名称，...，关节20名称]
	float64[] position：控制各关节运动至目标角度，格式为[关节1位置，关节2位置，...，关节20位置]，单位是弧度（rad），取值范围是[-π, π]

关节电机名称如下表

名称	描述
ANKLE	脚踝
KNEE	膝盖
BUTTOCK	臀部
WAIST	腰
NECK1	颈（水平）
NECK2	颈（俯仰）
LEFT_J1	左肩1
LEFT_J2	左肩2
LEFT_J3	左肩3
LEFT_J4	左肘
LEFT_J5	左腕1
LEFT_J6	左腕2
LEFT_J7	左腕3
RIGHT_J1	右肩1
RIGHT_J2	右肩2

RIGHT_J3	右肩3
RIGHT_J4	右肘
RIGHT_J5	右腕1
RIGHT_J6	右腕2
RIGHT_J7	右腕3

2. 高层全身协调与控制接口

高层全身协调与控制接口用于配置机器人运行参数（TCP坐标系/负载/速度）并执行笛卡尔空间运动（关节运动/直线运动/轨迹跟踪），调用内置运动学求解器（正/逆运动学）和切换高级模式（零力拖动/阻抗控制/遥操作）。

真机

Service	/control/set_property
生效版本	≥V0.3.0
接入方式	Service Call 基于 SetProperty.srv 的交互模式 —— 客户端发送请求，服务端返回响应
接口说明	动态配置机器人运行参数，支持速度比例、末端执行器位姿、错误清除、零力拖动使能、运行模式切换、阻抗参数设置等功能
接口参数	string property_type{ //操作类型，枚举 "SpeedRatio"; //设置机器人的运行速度比例 "TcpPrimary"; //设置左臂末端执行器的位置和姿态，用四元数表示 "TcpSecondary"; //设置右臂末端执行器的位置和姿态，用四元数表示 "ClearError"; //清除机器人错误状态 "SafeLevel"; //设置安全等级，统一调控机器人的碰撞检测、运动限制、力控阈值等一系列安全行为，数值越大，安全限制越严格 "EnableDrag"; //启用和关闭零力拖动功能 "EnableSleep"; //是否进入休眠（待机）状态 "QuickStop"; //快速停止 "SwitchRunMode"; //切换运行模式 "JointStiffness"; //在关节阻抗模式下设置关节刚度，数值越大，关节刚性越强。 "JointDamping"; //在关节阻抗模式下设置关节阻尼，数值越大，关节阻尼越大。 "CartStiffness"; //在笛卡尔阻抗模式下设置机器人在笛卡尔空间的刚度系数。

	"CartDampingRatio"; //在笛卡尔阻抗模式下设置机器人在笛卡尔空间的阻尼比。 }; string value{ //value的值与 property_type 关联 -若 property_type 为 SpeedRatio ， value 取值范围为 [0.01, 1.0] ， 数据类型是 float64， 示例: "value": "0.5"; -若 property_type 为 TcpPrimary ， value 的格式为 "x, y, z, qw, qx, qy, qz" ， 数据类型是 float64， 位置的单位是米， 姿态的单位是弧度， 用四元数表示， 示例: "value": "x, y, z, qw, qx, qy, qz "; -若 property_type 为 TcpSecondary ， value 的格式为 "x, y, z, qw, qx, qy, qz" ， 数据类型是 float64， 位置的单位是米， 姿态的单位是弧度， 用四元数表示， 示例: "value": "x, y, z, qw, qx, qy, qz "; -若 property_type 为 ClearError ， value 是空字符串， 示例: "value": " "; -若property_type为 SafeLevel， value 取值范围为[0.1, 1.0]， 数据类型是 float64， 示例: "value": "0.3"; -若 property_type 为 EnableDrag ， value 的数据类型是 bool， "1"表示启用零力拖动功能， "0"表示关闭零力拖动功能， 示例: "value": "1"; -若 property_type 为 EnableSleep， value 的数据类型是 bool， "1"表示启用休眠， "0"表示禁用休眠（唤醒）， 示例: "value": "1"; -若 property_type 为 QuickStop， value 的数据类型是 bool， "1"表示开启快速停止， "0"表示退出快速停止， 示例: "value": "1"; -若 property_type 为 SwitchRunMode， value 的数据类型是 double， "0"表示位置模式， "1"表示关节阻抗模式， "2"表示拖动模式， "3"表示笛卡尔阻抗模式； -若 property_type 为 JointStiffness， value 的数据类型是 float64， 单位是 牛米/弧度（Nm/rad） ， 示例： value: '50'; 也可以分别设置手臂关节每个关节的刚度大小, 数据数量必须为双臂自由度 14, 长度错误时无效， 示例： value: '50, 50, 10, 10, 10, 10, 10, 50, 50, 10, 10, 10, 10, 10'; -若 property_type 为 JointDamping， value 的数据类型是 float64， 单位是 牛米秒/弧度（Nm·s/rad） ， 示例： value: '50'， 必须与JointStiffness配合使用， 一起构成完整的阻抗控制； -若 property_type 为 CartStiffness， value 的数据类型是 float64， 单位是牛米/弧度（Nm/rad） ， 示例： value: '200'; -若 property_type 为 CartDampingRatio， value 的数据类型是 float64， 示例： value: '0.8'; } uint64 request_id: 返回此次指令的指令号， 用于追踪指令执行状态
--	---

Action	/control/move_robot
--------	---------------------

生效版本	≥V0.2
接入方式	Action Call 基于 MoveRobot.action 的交互模式 —— 客户端发起动作请求，服务端异步执行并反馈过程与结果
接口说明	控制机器人多模式运动，覆盖关节运动、直线运动、实时流控（关节 / 末端位姿），支持单臂 / 双臂 / 全身关节控制
接口参数	string type{ //设置运动运动控制模式，枚举 "MoveJoint"; //控制机器人各关节移动到指定角度，实现关节空间运动 "MoveLine"; //控制末端执行器沿直线运动到目标位姿 "StartJointStreaming"; //持续接收关节位置指令，动态调整关节状态 "StartEEPoseStreaming"; //持续接收末端位姿指令，动态调整末端位置 "StopMove"; //停止运动 }; string motion_group{ //指定参与运动控制的机器人部分，枚举 "Primary"; //左臂，关节数量为 7，适用于单臂独立任务 "Secondary"; //右臂，关节数量为 7，适用于单臂独立任务 "BothArms"; //双臂，关节数量为 14，适用于双臂协同任务 "Torso"; //躯干，关节数量为3 "WholeBody"; //全身关节, 关节数量为20，适用于全身协同运动 }; float32[] target{ //target 的值与 type 和 motion_group 关联 -若 type 为 MoveJoint，target 为关节角度数组，格式为 [关节1位置，…，关节n位置]，n 由 motion_group 决定，例如选择 Primary (左臂)时 n 为7, 选择 BothArms(双臂)时 n 为 14。角度的单位为弧度。 -若 type 为 MoveLine，target 为末端目标位姿或躯干位姿，末端目标位姿格式为 [x, y, z, qw, qx, qy, qz]，躯干位姿格式为[x, z, ry]，位置的单位是米，姿态的单位是弧度。 -若 type 为 StartJointStreaming / StartEEPoseStreaming，按对应控制逻辑，实时传入动态目标数据。 }; bool success: 返回布尔值，"1"表示此次指令成功执行，"0"表示此次指令执行失败 uint64 request_id: 返回此次指令的指令号，用于追踪指令执行过程、关联日志或状态查询

Topic	/feedback/robot_server_state
生效版本	≥V0.2
接入方式	订阅（subscribe）
接口说明	获取机器人服务器当前状态，包括运行模式、任务队列、关节与末端执行器位姿
接口参数	uint64 timestamp：时间戳 string status{ //枚举，server当前状态 Running; //运行中，可配合 latest_queued_id/latest_finished_id 追踪任务流程 Init; //初始状态，如果掉OP也会切换到此状态 Error; //异常，需结合 latest_errordrop_id 定位报错请求 Idle; //空闲中，无任务队列 }; string run_mode{ //枚举，机器人运行模式 Position; //位置模式 Drag; //拖动模式 Impedance; //阻抗模式 Sleep; //休眠模式(会下使能) }; string robot_statusword: 机器人状态字 "NOT_READY": 代表电机还未初始化成功 "READY": 代表电机已全OP且无报错, server会发送上使能指令 "ENABLED": 代表已上使能 "MOTOR_FAULT": 代表电机还未全OP或者有报错, 清错成功后切换到READY状态 uint64 latest_queued_id：最新入列的请求编号，表示当前已进入队列等待执行的最新请求ID，反映待执行任务的优先级与队列长度，若队列持续增长（latest_queued_id 频繁更新），需排查任务执行效率。 uint64 latest_finished_id：最新完成的请求编号，表示已经执行完成的最新请求ID，用于验证任务执行结果 uint64 latest_errordrop_id：最新报错指令编号，是故障排查的关键线索 string[] server_error_code: server报错码, 包含 "None": 无错误 "PositionOverLimit": 位置超限

"VelocityOverLimit": 速度超限 "TorqueOverLimit": 力矩超限
string[] motor_status: 20 个关节电机的运行状态, "OP"表示正常运行, 格式为["OP", "OP", ...]
float64[] motor_temperature: 20 个关节电机的温度, 格式为[33, 33, 33, ...], 单位℃
uint8[] motor_error_code: 电机错误码, 格式为[0, 0, 0, ...], 0: 无错误, 10: 速度误差超出限制值, 20: 位置误差超出限制值, 30: 主站掉线
float64[] joint_position: 机器人各关节的当前位置, 格式为 [-0.0015, -0.0015, -0.0011, ...], 单位为弧度 (rad)
float64[] joint_velocity: 机器人20个关节的关节速度, 格式为[0.0000, 0.0000, 0.0000, ...], 单位为弧度/秒
float64[] joint_torque: 机器人20个关节的实际关节力矩, 格式为[-2.0010, 1.3920, -2.6100, ...], 单位为Nm
float64[] joint_stiffness: 关节阻抗模式下的刚度
float64[] joint_damping: 关节阻抗模式下的阻尼
float64[] cart_stiffness: 笛卡尔阻抗模式下的刚度
float64[] joint_damping_ratio: 笛卡尔阻抗模式下的阻尼比
float64[] force_sensor: 六维力传感器数据, 格式为 [0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000], 前六位为左手, 后六位为右手
float64[] ee_pose: 末端执行器 (End-Effector) 的位姿, 用四元数表示, 格式为 [x, y, z, qw, qx, qy, qz, x, y, z, qw, qx, qy, qz], 前7位 [x, y, z, qw, qx, qy, qz]为左臂位姿, 后七位 [x, y, z, qw, qx, qy, qz]为右臂位姿, 位置的单位为米, 姿态的单位为弧度。
float64[] torso_pose_X_Z_RY: 躯干位姿[X坐标, Z坐标, RY旋转], 格式为[-0.0022, 1.1298, -0.0041], X/Z 坐标单位为米, RY 旋转单位为弧度

Topic	/control/streaming_command
生效版本	≥V0.2
接入方式	发布 (publish) 消息格式遵循 StreamingCommand.msg 的规范
接口说明	发送连续控制指令, 驱动机器人左右臂运动

接口参数	uint64 request_id：与 start streaming request 绑定的唯一 ID，用于指令标识,机器人服务器通过该 ID 区分不同流控任务，避免多任务指令混淆
	bool streaming_finished：标记当前指令是否为流控序列的最后一帧，True 表示发送后立即终止本次流控，机器人停止接收后续指令，可用于结束轨迹、复位手臂；False 表示持续激活流控模式，机器人等待下一帧指令，维持连续运动（如轨迹跟踪需高频发送多帧指令）。
	float32[] position：支持两种方式，需严格匹配 motion_group 配置。 • 双臂可达空间，直接指定关节角度，格式为[关节1位置，...，关节n位置]，数组长度由 motion_group 决定 • 通过位姿 [x, y, z, qw, qx, qy, qz] 控制末端执行器，数组长度与 motion_group 强关联，例如 motion_group 选择 Primary（左臂）时数组长度为7，选择 BothArms（双臂）时数组长度为14。位置的单位米，姿态的单位是弧度。

仿真环境

Service	/control/set_property_sim
生效版本	≥V0.3
接入方式	Service Call 基于 SetProperty.srv 的交互模式 —— 客户端发送请求，服务端返回响应
接口说明	动态配置机器人运行参数，支持速度比例、末端执行器位姿、错误清除、零力拖动使能等功能
接口参数	string property_type{ //操作类型，枚举 "SpeedRatio"; //设置机器人的运行速度比例 "TcpPrimary"; //设置左臂末端执行器的位置和姿态，用四元数表示 "TcpSecondary"; //设置右臂末端执行器的位置和姿态，用四元数表示 "ClearError"; //清除机器人错误状态 "SafeLevel"; //设置安全等级，统一调控机器人的碰撞检测、运动限制、力控阈值等一系列安全行为，数值越大，安全限制越严格 "EnableDrag"; //启用和关闭零力拖动功能 "EnableSleep"; //是否进入休眠（待机）状态 "SwitchRunMode"; //切换运行模式 "JointStiffness"; //在关节阻抗模式下设置关节刚度，数值越大，关节刚性越强。 "JointDamping"; //在关节阻抗模式下设置关节阻尼，数值越大，关节阻尼越大。 "CartStiffness"; //在笛卡尔阻抗模式下设置机器人在笛卡尔空间的刚度系数。

	"CartDampingRatio"; //在笛卡尔阻抗模式下设置机器人在笛卡尔空间的阻尼比。 }; string value{ //value的值与 property_type 关联 -若 property_type 为 SpeedRatio ， value 取值范围为 [0.01, 1.0] ， 数据类型是 float64， 示例: "value": "0.5"; -若 property_type 为 TcpPrimary ， value 的格式为 "x, y, z, qw, qx, qy, qz" ， 数据类型是 float64， 位置的单位是米， 姿态的单位是弧度， 用四元数表示， 示例: "value": "x, y, z, qw, qx, qy, qz "; -若 property_type 为 TcpSecondary ， value 的格式为 "x, y, z, qw, qx, qy, qz" ， 数据类型是 float64， 位置的单位是米， 姿态的单位是弧度， 用四元数表示， 示例: "value": "x, y, z, qw, qx, qy, qz "; -若 property_type 为 ClearError ， value 是空字符串， 示例: "value": " "; -若property_type为 SafeLevel， value 取值范围为[0.1, 1.0]， 数据类型是 float64， 示例: "value": "0.3"; -若 property_type 为 EnableDrag ， value 的数据类型是 bool， "1"表示启用零力拖动功能， "0"表示关闭零力拖动功能， 示例: "value": "1"; -若 property_type 为 EnableSleep， value 取值范围的数据类型是 bool， "1"表示启用休眠， "0"表示禁用休眠（唤醒）， 示例: "value": "1"; -若 property_type 为 SwitchRunMode， value 的数据类型是 double， "0"表示位置模式， "1"表示关节阻抗模式， "2"表示拖动模式， "3"表示笛卡尔阻抗模式； -若 property_type 为 JointStiffness， value 的数据类型是 float64， 单位是 牛米/弧度（Nm/rad） ， 示例： value: '50'； -若 property_type 为 JointDamping， value 的数据类型是 float64， 单位是 牛米秒/弧度（Nm·s/rad） ， 示例： value: '50'， 必须与JointStiffness配合使用， 一起构成完整的阻抗控制； -若 property_type 为 CartStiffness， value 的数据类型是 float64， 单位是牛米/弧度（Nm/rad） ， 示例： value: '200'； -若 property_type 为 CartDampingRatio， value 的数据类型是 float64， 示例： value: '0.8'； } uint64 request_id： 返回此次指令的指令号， 用于追踪指令执行状态
--	--

Action	/control/move_robot_sim
生效版本	≥V0.2
接入方式	Action Call

	基于 MoveRobot.action 的交互模式 —— 客户端发起动作请求，服务端异步执行并反馈过程与结果
接口说明	控制机器人多模式运动，覆盖关节运动、直线运动、实时流控（关节 / 末端位姿），支持单臂 / 双臂 / 全身关节控制
接口参数	string type{ //设置运动运动控制模式，枚举 "MoveJoint"; //控制机器人各关节移动到指定角度，实现关节空间运动 "MoveLine"; //控制末端执行器沿直线运动到目标位姿 "StartJointStreaming"; //持续接收关节位置指令，动态调整关节状态 "StartEEPoseStreaming"; //持续接收末端位姿指令，动态调整末端位置 "StopMove"; //停止运动 }; string motion_group{ //指定参与运动控制的机器人部分，枚举 "Primary"; //左臂，关节数量为 7，适用于单臂独立任务 "Secondary"; //右臂，关节数量为 7，适用于单臂独立任务 "BothArms"; //双臂，关节数量为 14，适用于双臂协同任务 "Torso"; //躯干，关节数量为3 "WholeBody"; //全身关节, 关节数量为20，适用于全身协同运动 }; float32[] target{ //target 的值与 type 和 motion_group 关联 -若 type 为 MoveJoint，target 为关节角度数组，格式为 [关节1位置，…，关节n位置]，n 由 motion_group 决定，例如选择 Primary (左臂)时 n 为7, 选择 BothArms(双臂)时 n 为 14。角度的单位为弧度。 -若 type 为 MoveLine，target 为末端目标位姿或躯干位姿，末端目标位姿格式为 [x, y, z, qw, qx, qy, qz]，躯干位姿格式为[x, z, ry]，位置的单位是米，姿态的单位是弧度。 -若 type 为 StartJointStreaming / StartEEPoseStreaming，按对应控制逻辑，实时传入动态目标数据。 }; bool success: 返回布尔值，"1"表示此次指令成功执行，"0"表示此次指令执行失败 uint64 request_id: 返回此次指令的指令号，用于追踪指令执行过程、关联日志或状态查询

Topic	/feedback/robot_server_state_sim
生效版本	≥V0.2

接入方式	订阅（subscribe）
接口说明	获取机器人服务器当前状态，包括运行模式、任务队列、关节与末端执行器位姿
接口参数	uint64 timestamp：时间戳
	string status{ //枚举，机器人服务器当前状态 Running; //运行中，可配合 latest_queued_id/latest_finished_id 追踪任务流程 Init; //初始状态，如果掉OP也会切换到此状态 Error; //异常，需结合 latest_errordrop_id 定位报错请求 Idle; //空闲中，无任务队列 };
	string run_mode{ //枚举，机器人运行模式 Position; //位置模式 Drag; //拖动模式 Impedance; //阻抗模式 Sleep; //休眠模式(会下使能) };
	uint64 latest_queued_id：最新入列的请求编号，表示当前已进入队列等待执行的最新请求ID，反映待执行任务的优先级与队列长度，若队列持续增长（latest_queued_id 频繁更新），需排查任务执行效率。
	uint64 latest_finished_id：最新完成的请求编号，表示已经执行完成的最新请求ID，用于验证任务执行结果
	uint64 latest_errordrop_id：最新报错指令编号，是故障排查的关键线索
	string[] motor_status：20 个关节电机的运行状态，"OP"表示正常运行，格式为["OP", "OP", ...]
	float64[] motor_temperature：20 个关节电机的温度，格式为[33, 33, 33, ...]，单位℃
	uint8[] motor_error_code：电机错误码，格式为[0, 0, 0, ...]，0：无错误，10：速度误差超出限制值，20：位置误差超出限制值，30：主站掉线
	float64[] joint_position：机器人各关节的当前位置，格式为 [-0.0015, -0.0015, -0.0011, ...]，单位为弧度（rad）
	float64[] joint_velocity：机器人20个关节的关节速度，格式为[0.0000, 0.0000, 0.0000, ...]，单位为弧度/秒

float64[] joint_torque: 机器人20个关节的实际关节力矩，格式为[-2.0010, 1.3920, -2.6100, ...]，单位为Nm
float64[] force_sensor: 六维力传感器数据，格式为[0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000,0.0000]，前六位为左手, 后六位为右手
float64[] ee_pose: 末端执行器（End-Effector）的位姿，用四元数表示，格式为 [x, y, z, qw, qx, qy, qz, x, y, z, qw, qx, qy, qz]，前7位 [x, y, z, qw, qx, qy, qz]为左臂位姿，后七位 [x, y, z, qw, qx, qy, qz]为右臂位姿，位置的单位为米，姿态的单位为弧度。
float64[] torso_pose_X_Z_RY: 躯干位姿[X坐标, Z坐标, RY旋转]，格式为[-0.0022, 1.1298, -0.0041]，X/Z 坐标单位为米，RY 旋转单位为弧度

Topic	/control/streaming_command_sim
生效版本	≥V0.2
接入方式	发布（publish） 消息格式遵循 StreamingCommand.msg 的规范
接口说明	发送连续控制指令，驱动机器人左右臂运动
接口参数	uint64 request_id: 与 start streaming request 绑定的唯一 ID，用于指令标识,机器人服务器通过该 ID 区分不同流控任务，避免多任务指令混淆 bool streaming_finished: 标记当前指令是否为流控序列的最后一帧，True 表示发送后立即终止本次流控，机器人停止接收后续指令，可用于结束轨迹、复位手臂；False 表示持续激活流控模式，机器人等待下一帧指令，维持连续运动（如轨迹跟踪需高频发送多帧指令）。 float32[] position: 支持两种方式，需严格匹配 motion_group 配置。 • 双臂可达空间范围内，直接指定关节角度，按照 motion_group 的关节数量和顺序，为每个关节设置角度值，驱动各个关节运动，格式为[关节1位置，...，关节n位置]，数组长度由 motion_group 决定 • 通过位姿 [x, y, z, qw, qx, qy, qz] 控制末端执行器，数组长度与 motion_group 强关联，例如 motion_group 选择 Primary（左臂）时数组长度为7，选择 BothArms（双臂）时数组长度为14。位置的单位米，姿态的单位是弧度。

3. 末端执行器接口

末端执行器接口用于操作末端执行器（灵巧手指节控制/二指夹爪开合力控）。

真机

Topic	/feedback/ee/left
生效版本	≥V0.2
接入方式	订阅（subscribe） 消息类型：end_effector_interfaces/msg/EEFeedback
接口说明	实时获取左侧末端执行器的状态，包含位置、速度、电流、力等信息
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息 string ee_name { //末端设备名称（品牌_型号_版本） BrainCo_Revo1_T; //强脑进口版本 BrainCo_Revo1_E; //强脑国产版本 Linker_L6; //灵心巧手L6 Linker_L20; //灵心巧手L20 PaXini_Dex_H13; //帕西尼H13 DH_PGC_140_50; //大寰夹爪 DH_AG_160_95; //大寰夹爪 } uint8[] status：末端执行器状态，0：IDLE（空闲 / 到位停止），1：RUNNING（运动中），2：STALL（堵转 / 保护停止），100：UNCONNECT（USB未连接），255：UNKNOWN（找不到末端设备） EEJointState[] joint_states：关节状态 { string name: 关节名称, 6自由度灵巧手：T_MCP(拇指弯曲),T_CMC_YAW(拇指摆动),IF_MCP_PITCH(食指弯曲),MF_MCP_PITCH(中指弯曲),RF_MCP_PITCH(无名指弯曲),LF_MCP_PITCH(小指弯曲) 二指夹：FINGER1（夹爪开合） # 关节位置 [0,100]，单位为百分比（%） float32 position # 关节速度 [-100,100]，单位为百分比每秒（%/s） float32 velocity # 关节电流 [-100,100]，单位为百分比（%） float32 current # 关节温度 单位为摄氏度 float32 temperature # 关节状态码 0：IDLE 1：RUNNING 2：STALL 100：UNCONNECT 255：UNKNOWN uint8 state # 关节故障码，强脑，大寰夹爪无故障码，其他设备请参考用户手册 uint8 error_code

	<pre>} EETactileState[] tactile_states: 触觉传感器状态 { # 区域名称, 与关节名称一致 string field_name # 阵列数据 geometry_msgs/Wrench[] distributed_datas # 单位 N # 阵列行数 uint32 row # 阵列列数 uint32 col }</pre>
--	--

Topic	/feedback/ee/right
生效版本	≥V0.2
接入方式	订阅 (subscribe) 消息类型: end_effector_interfaces/msg/EEFeedback
接口说明	实时获取右侧末端执行器的状态, 包含位置、速度、电流、力等信息
接口参数	header: 标准消息头, 包含时间戳 (stamp)、坐标系 (frame_id) 等元信息 <pre>string ee_name { //末端设备名称 (品牌_型号_版本) BrainCo_Revo1_T; //强脑进口版本 BrainCo_Revo1_E; //强脑国产版本 Linker_L6; //灵心巧手L6 Linker_L20; //灵心巧手L20 PaXini_Dex_H13; //帕西尼H13 DH_PGC_140_50; //大寰夹爪 DH_AG_160_95; //大寰夹爪 }</pre> uint8[] status: 末端执行器状态, 0: IDLE (空闲 / 到位停止), 1: RUNNING (运动中), 2: STALL (堵转 / 保护停止), 100: UNCONNECT (USB未连接), 255: UNKNOWN (找不到末端设备) EEJointState[] joint_states: 关节状态 <pre>{</pre>

	<pre>string name: 关节名称, 6自由度灵巧手: T_MCP(拇指弯曲),T_CMC_YAW(拇指摆动),IF_MCP_PITCH(食指弯曲),MF_MCP_PITCH(中指弯曲),RF_MCP_PITCH(无名指弯曲),LF_MCP_PITCH(小指弯曲) 二指夹: FINGER1 (夹爪开合) # 关节位置 [0,100], 单位为百分比 (%) float32 position # 关节速度 [-100,100], 单位为百分比每秒 (%/s) float32 velocity # 关节电流 [-100,100], 单位为百分比 (%) float32 current # 关节温度 单位为摄氏度 float32 temperature # 关节状态码 0: IDLE 1: RUNNING 2: STALL 100: UNCONNECT 255: UNKNOWN uint8 state # 关节故障码, 强脑, 大寰夹爪无故障码, 其他设备请参考用户手册 uint8 error_code }</pre>
	<pre>EEtactileState[] tactile_states: 触觉传感器状态 { # 区域名称, 与关节名称一致 string field_name # 阵列数据 geometry_msgs/Wrench[] distributed_datas # 单位 N # 阵列行数 uint32 row # 阵列列数 uint32 col }</pre>

Topic	/control/ee/left
生效版本	≥V0.2
接入方式	发布 (publish) 消息类型: end_effector_interfaces/msg/EEJointControl.msg
接口说明	发布指令, 控制左侧末端执行器的状态
接口参数	header: 标准消息头, 包含时间戳 (stamp)、坐标系 (frame_id) 等元信息 string[] joint_names{ //关节名称列表 // 6自由度灵巧手:

	<pre>“T_MCP” ;//拇指弯曲 “T_CMC_YAW” ;//拇指摆动 “IF_MCP_PITCH” ;//食指弯曲 “MF_MCP_PITCH” ;//中指弯曲 “RF_MCP_PITCH” ;//无名指弯曲 “LF_MCP_PITCH” ;//小指弯曲 // 夹爪 “FINGER1” ;// 夹爪开合 };</pre>
	<pre>uint8 mode{ //控制模式，枚举 0: POSITION; //位置控制 1: VELOCITY; //速度控制 2: CURRENT; // 电流控制 3: FORCE; // 力控制 };</pre>
	float32[] value: 格式为数组 [q1,q2,q3,q4,q5,q6]，关节数量由 name 决定，与 name 一一对应，默认 6 个，含义由 mode 决定，单位为百分比 (%) • 当 mode =0时 value 为各关节的 “弯曲程度比例”，取值范围是[0.0, 100.0]; • 当 mode =1时 value 为各关节的 “运动速度比例”，取值范围是[-100, 100]; • 当 mode =2时 value 为各关节的 “电流输出比例”，取值范围是[-100, 100]; • 当 mode =3时 value 为各关节的 “力输出比例”，取值范围是[-100, 100]

Topic	/control/ee/right
生效版本	≥V0.2
接入方式	发布（publish） 消息类型：end_effector_interfaces/msg/EEJointControl.msg
接口说明	发布指令，控制右侧末端执行器的状态
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息 string[] joint_names{ //关节名称列表 // 6自由度灵巧手： “T_MCP” ;//拇指弯曲 “T_CMC_YAW” ;//拇指摆动

	<pre>“IF_MCP_PITCH” ;//食指弯曲 “MF_MCP_PITCH” ;//中指弯曲 “RF_MCP_PITCH” ;//无名指弯曲 “LF_MCP_PITCH” ;//小指弯曲 // 夹爪 “FINGER1” ;// 夹爪开合 };</pre>
	<pre>uint8 mode{ //控制模式，枚举 0: POSITION; //位置控制 1: VELOCITY; //速度控制 2: CURRENT; // 电流控制 3: Force; // 力控制 };</pre>
	float32[] value：格式为数组 [q1,q2,q3,q4,q5,q6]，关节数量由 <code>name</code> 决定，与 <code>name</code> 一一对应，默认 6 个，含义由 <code>mode</code> 决定，单位为百分比（%） • 当 <code>mode</code> =0时 <code>value</code> 为各关节的 “弯曲程度比例”，取值范围是[0.0, 100.0]； • 当 <code>mode</code> =1时 <code>value</code> 为各关节的 “运动速度比例”，取值范围是[-100, 100]； • 当 <code>mode</code> =2时 <code>value</code> 为各关节的 “电流输出比例”，取值范围是[-100, 100]； • 当 <code>mode</code> =3时 <code>value</code> 为各关节的 “力输出比例”，取值范围是[-100, 100]

仿真环境

Topic	/feedback_sim/ee/left
生效版本	≥V0.2
接入方式	订阅（subscribe） 消息类型：end_effector_interfaces/msg/EEFeedback
接口说明	实时获取左侧末端执行器的状态，包含位置、速度、电流、力等信息
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息 string ee_name{ //末端设备名称（品牌_型号_版本） BrainCo_Revo1_T; //强脑进口版本 BrainCo_Revo1_E; //强脑国产版本 Linker_L6; //灵心巧手L6

```
Linker_L20; //灵心巧手L20
PaXini_Dex_H13; //帕西尼H13
DH_PGC_140_50; //大寰夹爪
DH_AG_160_95; //大寰夹爪
}
```

uint8[] status: 末端执行器状态, 0: IDLE (空闲 / 到位停止), 1: RUNNING (运动中), 2: STALL (堵转 / 保护停止), 100: UNCONNECT (USB未连接), 255: UNKNOWN (找不到末端设备)

EEJointState[] joint_states: 关节状态,

```
{
string name: 关节名称, 6自由度灵巧手: T_MCP(拇指弯曲), T_CMC_YAW(拇指摆动), IF_MCP_PITCH(食指弯曲), MF_MCP_PITCH(中指弯曲), RF_MCP_PITCH(无名指弯曲), LF_MCP_PITCH(小指弯曲)

二指夹: FINGER1
# 关节位置 [0,100], 单位为百分比 (%)
float32 position
# 关节速度 [-100,100], 单位为百分比每秒 (%/s)
float32 velocity
# 关节电流 [-100,100], 单位为百分比 (%)
float32 current
# 关节温度 单位为摄氏度
float32 temperature
# 关节状态码 0: IDLE 1: RUNNING 2: STALL 100: UNCONNECT 255: UNKNOWN
uint8 state
# 关节故障码, 强脑, 大寰夹爪无故障码, 其他设备请参考用户手册
uint8 error_code
}
```

EETactileState[] tactile_states: 触觉传感器状态,

```
{
# 区域名称, 与关节名称一致
string field_name
# 阵列数据
geometry_msgs/Wrench[] distributed_datas # 单位 N
# 阵列行数
uint32 row
# 阵列列数
uint32 col
}
```

Topic	/feedback_sim/ee/right
生效版本	≥V0.2
接入方式	订阅（subscribe）
接口说明	实时获取右侧末端执行器的状态，包含位置、速度、电流、力等信息
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息 string ee_name { //末端设备名称（品牌_型号_版本） BrainCo_Revo1_T; //强脑进口版本 BrainCo_Revo1_E; //强脑国产版本 Linker_L6; //灵心巧手L6 Linker_L20; //灵心巧手L20 PaXini_Dex_H13; //帕西尼H13 DH_PGC_140_50; //大寰夹爪 DH_AG_160_95; //大寰夹爪 } uint8[] status：末端执行器状态，0：IDLE（空闲 / 到位停止），1：RUNNING（运动中），2：STALL（堵转 / 保护停止），100：UNCONNECT（USB未连接），255：UNKNOWN (找不到末端设备)

EEJointState[] joint_states: 关节状态，

```
{
string name: 关节名称, 6自由度灵巧手: T_MCP(拇指弯曲),T_CMC_YAW(拇指摆动),IF_MCP_PITCH(食指弯曲),MF_MCP_PITCH(中指弯曲),RF_MCP_PITCH(无名指弯曲),LF_MCP_PITCH(小指弯曲)

二指夹: FINGER1
# 关节位置 [0,100] , 单位为百分比 (%)
float32 position
# 关节速度 [-100,100], 单位为百分比每秒 (%/s)
float32 velocity
# 关节电流 [-100,100], 单位为百分比 (%)
float32 current
# 关节温度 单位为摄氏度
float32 temperature
# 关节状态码 0: IDLE 1: RUNNING 2: STALL 100: UNCONNECT 255: UNKNOWN
uint8 state
# 关节故障码, 强脑, 大寰夹爪无故障码, 其他设备请参考用户手册
uint8 error_code
}
```

EETactileState[] tactile_states: 触觉传感器状态，

```
{
# 区域名称, 与关节名称一致
string field_name
# 阵列数据
geometry_msgs/Wrench[] distributed_datas # 单位 N
# 阵列行数
uint32 row
# 阵列列数
uint32 col
}
```

Topic	/control_sim/ee/left
生效版本	≥V0.2
接入方式	发布（publish） 消息类型：end_effector_interfaces/msg/EEJointControl.msg
接口说明	发布指令，控制左侧末端执行器的状态
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息

	<pre>string[] joint_names{ //关节名称列表 // 6自由度灵巧手： “T_MCP” ;//拇指弯曲 “T_CMC_YAW” ;//拇指摆动 “IF_MCP_PITCH” ;//食指弯曲 “MF_MCP_PITCH” ;//中指弯曲 “RF_MCP_PITCH” ;//无名指弯曲 “LF_MCP_PITCH” ;//小指弯曲 // 夹爪 “FINGER1” ; };</pre>
	<pre>uint8 mode{ //控制模式，枚举 0: POSITION; //位置控制 1: VELOCITY; //速度控制 2: CURRENT; // 电流控制 3: FORCE; // 力控制 };</pre>
	float32[] value: 格式为数组 [q1,q2,q3,q4,q5,q6], 关节数量由 <code>name</code> 决定, 与 <code>name</code> 一一对应, 默认 6 个, 含义由 <code>mode</code> 决定, 单位为百分比 (%) • 当 <code>mode</code> =0时 <code>value</code> 为各关节的 “弯曲程度比例”, 取值范围是[0.0, 100.0]; • 当 <code>mode</code> =1时 <code>value</code> 为各关节的 “运动速度比例”, 取值范围是[-100, 100]; • 当 <code>mode</code> =2时 <code>value</code> 为各关节的 “电流输出比例”, 取值范围是[-100, 100]; • 当 <code>mode</code> =3时 <code>value</code> 为各关节的 “力输出比例”, 取值范围是[-100, 100]

Topic	/control_sim/ee/right
生效版本	≥V0.2
接入方式	发布 (publish) 消息类型: end_effector_interfaces/msg/EEJointControl.msg
接口说明	发布指令, 控制右侧末端执行器的状态
接口参数	header: 标准消息头, 包含时间戳 (stamp)、坐标系 (frame_id) 等元信息 string[] joint_names{ //关节名称列表

	<pre>// 6自由度灵巧手： “T_MCP” ; //拇指弯曲 “T_CMC_YAW” ; //拇指摆动 “IF_MCP_PITCH” ; //食指弯曲 “MF_MCP_PITCH” ; //中指弯曲 “RF_MCP_PITCH” ; //无名指弯曲 “LF_MCP_PITCH” ; //小指弯曲 // 夹爪 “FINGER1” ; };</pre>
	<pre>uint8 mode{ //控制模式，枚举 0: POSITION; //位置控制 1: VELOCITY; //速度控制 2: CURRENT; // 电流控制 3: FORCE; // 力控制 };</pre>
	float32[] value: 格式为数组 [q1,q2,q3,q4,q5,q6]，关节数量由 <code>name</code> 决定，与 <code>name</code> 一一对应，默认 6 个，含义由 <code>mode</code> 决定，单位为百分比 (%) - 当 <code>mode</code> =0时 <code>value</code> 为各关节的 “弯曲程度比例”，取值范围是[0.0, 100.0]; - 当 <code>mode</code> =1时 <code>value</code> 为各关节的 “运动速度比例”，取值范围是[-100, 100]; - 当 <code>mode</code> =2时 <code>value</code> 为各关节的 “电流输出比例”，取值范围是[-100, 100]; - 当 <code>mode</code> =3时 <code>value</code> 为各关节的 “力输出比例”，取值范围是[-100, 100]

4. 底盘控制接口

(1) 建图

Service	/mapping_node/set_map_name
生效版本	≥V0.3.2
接入方式	Service Call 自定义ROS2消息格式：nav_interfaces/srv/SetMapName
接口说明	设置地图名称（必须在 <code>start_mapping</code> 前调用）
接口参数	请求：

	String map_name：在建图开始前，设置本次建图输出的地图文件名，非空字符串，格式为英文 / 数字 / 下划线
	响应：
	Bool success：地图名称是否设置成功，取值为1表示设置成功，取值为1表示设置失败； String message：返回的结果描述信息

Service	/mapping_node/start_mapping
生效版本	≥V0.3.2
接入方式	Service Call 标准ROS2消息格式 std_srvs/srv/Trigger
接口说明	启动建图
接口参数	请求：空请求
	响应： Bool success：启动建图是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

Service	/mapping_node/finish_mapping
生效版本	≥V0.3.2
接入方式	Service Call 标准ROS2消息格式 std_srvs/srv/Trigger
接口说明	完成建图
接口参数	请求：空请求
	响应： Bool success：是否结束建图并保存成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

(2) 定位

Service	/localization_node/start_localization
---------	---------------------------------------

生效版本	≥V0.3.2
接入方式	Service Call 标准ROS2消息格式 std_srvs/srv/Trigger
接口说明	启动定位（开机自动触发）
接口参数	请求：空请求 响应： Bool success：定位启动是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

Service	/localization_node/stop_localization
生效版本	≥V0.3.2
接入方式	Service Call 标准ROS2消息格式 std_srvs/srv/Trigger
接口说明	停止定位
接口参数	请求：空请求 响应： Bool success：定位停止是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

Service	/localization_node/reset_localization
生效版本	≥V0.3.2
接入方式	Service Call 标准ROS2消息格式 std_srvs/srv/Trigger
接口说明	重置定位状态（先关闭当前定位，再重新启动一次定位）
接口参数	请求：空请求 响应： Bool success：定位重置是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

Service	/localization_node/relocalization
生效版本	≥V0.3.2
接入方式	Service Call 自定义ROS2消息格式 nav_interfaces/srv/SetPose
接口说明	触发重定位（给定一个初始位姿，触发定位模块基于该初始位姿进行重定位）
接口参数	请求： float64[] pose.position：底盘在地图中的位置坐标，格式为[x,y,z]（笛卡尔坐标系下的位置矩阵），单位为米，根据地图坐标系取值； float64[] pose.orientation：底盘的姿态，用四元数表示，格式为[x,y,z,w]（四元数，满足$x^2+y^2+z^2+w^2=1$）； pose.covariance：位姿协方差矩阵，表示初始位姿的置信度，对角线元素越大，置信度越低，通常初始重定位可设为较大值（如[0.1, 0, ..., 0.1]） 响应： Bool success：重定位是否成功触发，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

（3）地图管理

Service	/map_manager/get_map_list
生效版本	≥V0.3.2
接入方式	Service Call 自定义ROS2消息格式 nav_interfaces/srv/GetMapList
接口说明	获取当前地图目录中所有有效的地图
接口参数	请求：空请求 响应： String[] map_list：所有有效地图的名称列表，空数组表示无有效地图； Bool success：查询是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

Service	/map_manager/get_map_folder
---------	-----------------------------

生效版本	≥V0.3.2
接入方式	Service Call 自定义ROS2消息格式 nav_interfaces/srv/GetString
接口说明	获取当前地图的存储目录
接口参数	请求：空请求 响应： String data：地图保存的绝对 / 相对路径 Bool success：查询是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

Service	/map_manager/get_map_name
生效版本	≥V0.3.2
接入方式	Service Call 自定义ROS2消息格式 nav_interfaces/srv/GetString
接口说明	获取当前定位中加载的地图名称
接口参数	请求：空请求 响应： String data：当前加载的地图名称 Bool success：查询是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

Service	/map_manager/get_current_map_info
生效版本	≥V0.3.2
接入方式	Service Call 自定义ROS2消息格式 nav_interfaces/srv/GetMapInfo
接口说明	获取当前加载地图的信息
接口参数	请求：空请求 响应：

	map_info：地图元数据，包含map_load_time、resolution、width、height、origin子字段； String map_name：当前地图名称； String map_path：当前地图文件的完整路径； Bool success：查询是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息
--	---

Service	/map_manager/load_map
生效版本	≥V0.3.2
接入方式	Service Call 自定义ROS2消息格式 nav_interfaces/srv/LoadMap
接口说明	加载指定名称的地图
接口参数	请求： String map_name：要加载的地图名称（必须是 <code>get_map_list</code> 返回的有效地图名称） 响应： Bool success：加载是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

Service	/map_manager/remove_map
生效版本	≥V0.3.2
接入方式	Service Call 自定义ROS2消息格式 nav_interfaces/srv/DeleteMap
接口说明	删除指定名称的地图
接口参数	请求： String map_name：要删除的地图名称（必须是 <code>get_map_list</code> 返回的有效地图名称） 响应： Bool success：删除是否成功，取值为1表示成功，取值为0表示失败； String message：返回的结果描述信息

(4) 差速底盘控制接口

消息格式遵循 geometry_msgs/Twist 标准

Topic	/feedback/odom
生效版本	≥V0.3
接入方式	订阅 (subscribe) 标准ROS2消息 nav_msgs/msg/Odometry
接口说明	获取机器人当前里程计信息 (Odometry)，包含机器人位姿与运动速度信息。 速度信息位于：msg.twist.twist
接口参数	float64 twist.twist.linear.x：获取 X 轴方向上的线速度，单位为 m/s，取值范围为 [-1.0, 1.0]
	float64 twist.twist.angular.z：获取绕 Z 轴的角速度，单位为 rad/s，取值范围为 [-1.0, 1.0]

Topic	/control/cmd_vel
生效版本	≥V0.2
接入方式	发布 (publish)
接口说明	发布指令，控制机器人的速度，通过设置线速度、角速度，控制机器人运动轨迹、姿态调整
接口参数	float64 linear.x：获取 X 轴方向上的线速度，单位为 m/s，取值范围为 [-1.0, 1.0]
	float64 angular.z：获取绕 Z 轴的角速度，单位为 rad/s，取值范围为 [-1.0, 1.0]

(5) 全向移动底盘控制接口

全向移动底盘控制接口用于实时获取底盘 8 个电机的运行状态与指令下发。

Topic	/feedback/omni_chassis_motors
生效版本	≥V0.2
接入方式	订阅 (subscribe)
接口说明	实时获取全向底盘的电机速度和转向电机角度
接口参数	header：标准消息头，包含时间戳 (stamp)、坐标系 (frame_id) 等元信息

	float64[] position：全向底盘的转向电机角度，格式为[左前轮角度, 右前轮角度, 左后轮角度, 右后轮角度]，取值范围是 $[-\pi/2, \pi/2]$ ，单位是弧度（rad）
	float64[] speed：全向底盘的电机速度，格式为[左前轮线速度, 右前轮线速度, 左后轮线速度, 右后轮线速度]，取值范围是 $[-1.0, 1.0]$ ，单位是米/秒（m/s）

Topic	/control/omni_chassis
生效版本	≥V0.2
接入方式	发布（publish）
接口说明	发布指令，实现全向底盘移动
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息
	float64[] position：全向底盘的转向电机角度，格式为[左前轮角度, 右前轮角度, 左后轮角度, 右后轮角度]，取值范围是 $[-\pi/2, \pi/2]$ ，单位是弧度（rad）
	float64[] speed：全向底盘的电机速度，格式为[左前轮线速度, 右前轮线速度, 左后轮线速度, 右后轮线速度]，取值范围是 $[-1.0, 1.0]$ ，单位是米/秒（m/s）

(6) 底盘位置接口

Topic	/feedback/chassis_pose
生效版本	≥V0.2
接入方式	订阅（subscribe） 基于标准ROS2 消息 geometry_msgs/PoseStamped 的交互
接口说明	实时获取底盘在地图中的位姿（需开启导航功能）
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息
	float64[] pose.position：底盘在地图中的位置坐标，格式为[x,y,z]（笛卡尔坐标系下的位置矩阵），单位为 m
	float64[] pose.orientation：底盘的姿态，格式为[x,y,z,w]（四元数，满足 $x^2+y^2+z^2+w^2=1$ ）

Topic	/control/nav2_pose
生效版本	≥V0.2
接入方式	发布（publish）

	基于标准ROS2 消息 geometry_msgs/PoseStamped 的交互
接口说明	发布指令，控制底盘移动到目标位置（需开启导航功能），到达目标位置时的姿态精度相对较低，带自主避障能力。
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息
	float64[] pose.position：底盘在地图中的目标位置，格式为[x,y,z]（笛卡尔坐标系下的位置矩阵），单位为 m
	float64[] pose.orientation：底盘的姿态，格式为[x,y,z,w]（四元数，满足 $x^2+y^2+z^2+w^2=1$ ）

Topic	/control/ctrl_pose
生效版本	≥V0.2
接入方式	发布（publish） 标准ROS2消息 geometry_msgs/PoseStamped
接口说明	发布指令，控制底盘移动到目标位置（需开启导航功能），仅支持直线移动和原地旋转，不包含自主避障能力，坐标系原点为当前点；
接口参数	header：标准消息头
	float64[] pose.position：目标位置的笛卡尔坐标（X/Y/Z 轴的位置），格式为[x, y, z]，范围根据机器人工作空间（地图）确定，单位为m
	float64[] pose.orientation：目标姿态的四元数表示（用于描述机器人的空间朝向），格式为[x, y, z, w]，需满足四元数约束： $x^2+y^2+z^2+w^2=1$

(7) 底盘基础服务接口

Service	/save_charging_pose
生效版本	≥V0.3
接入方式	Service Call 基于ROS标准std_srvs/srv/Empty格式——客户端发送请求，服务端返回响应
接口说明	保存回充点，将机器人当前所在位置设置为自动回充的基准点（充电桩正前方0.7m-1m）
接口参数	请求：空请求，通常发送 {} 或空字符串触发。
	响应： Bool success：位置保存是否成功，“1”表示成功，“0”表示不成功。

	String message：返回的提示信息，通常包含保存的回充点坐标信息。
--	--

Action	/auto_charing
生效版本	≥V0.3
接入方式	Action Call
接口说明	控制机器人自动寻找并对接充电桩，通过该接口下发回充指令，并实时获取机器人的回充状态。
接口参数	请求：
	Object goal：下发指令，示例：{force:true}，强制回充
	响应：
	String feedback：反馈当前状态，包括运行中 / 成功 / 失败 Bool result：回充最终是否成功

5. 全方位感知系统接口

全方位感知系统接口用于获取传感器原始数据（双目/底盘相机图像流、多部位IMU、激光雷达点云）。

双目相机

使用标准的 sensor_msgs/Image 消息类型

Topic	/camera/left_eye
生效版本	≥V0.2
接入方式	订阅（subscribe） 基于标准的ROS2 消息 sensor_msgs/Image 的交互
接口说明	实时获取双目相机左目的图像数据及参数
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， frame_id= "left_eye" (左目)
	uint32 height：图像高度（行数）
	uint32 width：图像宽度（列数）
	string encoding：图像编码格式

uint8 is_bigendian：字节序标识，0 表示小端序，1 表示大端序，通常取值为 0
uint32 step：每行字节数，step = width * 每个像素字节数
uint8[] data：图像数据本体（按行存储），数据长度 = height * step

Topic	/camera/right_eye
生效版本	≥V0.2
接入方式	订阅（subscribe） 基于标准的ROS2 消息 sensor_msgs/Image 的交互
接口说明	实时获取双目相机右目的图像数据及参数
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， frame_id= “right_eye” (右目)
	uint32 height：图像高度（行数）
	uint32 width：图像宽度（列数）
	string encoding：图像编码格式
	uint8 is_bigendian：字节序标识，0 表示小端序，1 表示大端序，通常取值为 0
	uint32 step：每行字节数，step = width * 每个像素字节数
	uint8[] data：图像数据本体（按行存储），数据长度 = height * step

Topic	/camera/kfc_compressed
生效版本	≥V0.2
接入方式	订阅（subscribe） 基于标准的ROS2 消息 sensor_msgs/CompressedImage 的交互
接口说明	实时获取双目相机的压缩图像
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id= “kfc_compressed”
	string format：指定图像的压缩格式，固定为"jpeg"
	uint8[] data：存储JPEG 格式压缩后的图像字节数据（JPEG 文件的字节流）

Topic	/camera/left_eye_resize
生效版本	≥V0.2
接入方式	订阅（subscribe） 基于标准的ROS2 消息 sensor_msgs/Image 的交互
接口说明	实时获取经过尺寸调整后的左目图像
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息，header.frame_id= “left_eye_resize”
	uint32 height：图像高度（行数）
	uint32 width：图像宽度（列数）
	string encoding：图像编码格式
	uint8 is_bigendian：字节序标识，0 表示小端序，1 表示大端序，通常取值为 0
	uint32 step：每行字节数，step = width * 每个像素字节数
	uint8[] data：图像数据本体（按行存储），数据长度 = height * step

Topic	/camera/right_eye_resize
生效版本	≥V0.2
接入方式	订阅（subscribe） 基于标准的ROS2 消息 sensor_msgs/Image 的交互
接口说明	实时获取经过尺寸调整后的右目图像
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息，header.frame_id= “right_eye_resize”
	uint32 height：图像高度（行数）
	uint32 width：图像宽度（列数）
	string encoding：图像编码格式
	uint8 is_bigendian：字节序标识，0 表示小端序，1 表示大端序，通常取值为 0
	uint32 step：每行字节数，step = width * 每个像素字节数
	uint8[] data：图像数据本体（按行存储），数据长度 = height * step

Topic	/camera/kfc_calib_data
生效版本	≥V0.3
接入方式	订阅（subscribe） 使用ROS2标准std_msgs/String消息格式
接口说明	实时获取相机内参信息
接口参数	string data：相机内参信息，格式为 yaml ，包含包含相机的焦距、主点、畸变系数等关键参数，以及相机的标定参数。

前后底盘相机

使用ROS标准消息结构

Topic	/orbbec/front/rgb
生效版本	≥V0.3
接入方式	订阅（subscribe） 使用ROS标准sensor_msgs/Image消息格式
接口说明	获取前底盘相机的彩色RGB 图
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id="chassis_camera_front_link" (front)

Topic	/orbbec/front/depth
生效版本	≥V0.3
接入方式	订阅（subscribe） 使用ROS标准sensor_msgs/Image消息格式
接口说明	获取前底盘相机的深度图
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id="chassis_depth_front_link" (front)

Topic	/orbbec/front/camera_info

生效版本	≥V0.3
接入方式	订阅（subscribe） 使用ROS标准sensor_msgs/CameraInfo消息格式
接口说明	获取前底盘相机的相机内参、畸变信息
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id="chassis_camera_front_link" (front)

Topic	/orbbec/rear/rgb
生效版本	≥V0.3
接入方式	订阅（subscribe） 使用ROS2标准sensor_msgs/Image消息格式
接口说明	获取后底盘相机的彩色RGB 图
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id="chassis camera_rear_link" (rear)

Topic	/orbbec/rear/depth
生效版本	≥V0.3
接入方式	订阅（subscribe） 使用ROS2标准sensor_msgs/Image消息格式
接口说明	获取后底盘相机的深度图
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id="chassis_depth_rear_link" (rear)

Topic	/orbbec/rear/camera_info
生效版本	≥V0.3
接入方式	订阅（subscribe） 使用ROS2的标准sensor_msgs/CameraInfo消息格式
接口说明	获取后底盘相机的相机内参、畸变信息

接口参数	header: 标准消息头, 包含时间戳 (stamp)、坐标系 (frame_id) 等元信息, header.frame_id="chassis_camera_rear_link" (rear)
------	--

腕部相机

使用标准的ros2消息sensor_msgs/Image

Topic	/camera_l/color/image_rect_raw
生效版本	≥V0.3.0
接入方式	订阅 (subscribe) 基于标准的ROS2 消息 sensor_msgs/Image 的交互
接口说明	实时获取左手腕部相机的彩色 RGB 图像
接口参数	header: 标准消息头, 包含时间戳 (stamp)、坐标系 (frame_id) 等元信息
	uint32 height: 图像高度 (行数)
	uint32 width: 图像宽度 (列数)
	string encoding: 图像编码格式
	uint8 is_bigendian: 字节序标识, 0 表示小端序, 1 表示大端序 , 通常取值为 0
	uint32 step: 每行字节数, step = width * 每个像素字节数
	uint8[] data: 图像数据本体 (按行存储) , 数据长度 = height * step

Topic	/camera_l/aligned_depth_to_color/image_raw
生效版本	≥V0.3.0
接入方式	订阅 (subscribe) 基于标准的ROS2 消息 sensor_msgs/Image 的交互
接口说明	实时获取左手腕部相机的深度图
接口参数	header: 标准消息头, 包含时间戳 (stamp)、坐标系 (frame_id) 等元信息
	uint32 height: 图像高度 (行数)
	uint32 width: 图像宽度 (列数)
	string encoding: 图像编码格式

uint8 is_bigendian：字节序标识，0 表示小端序，1 表示大端序 ，通常取值为 0
uint32 step：每行字节数，step = width * 每个像素字节数
uint8[] data：图像数据本体（按行存储），数据长度 = height * step

Topic	/camera_r/color/image_rect_raw
生效版本	≥V0.3.0
接入方式	订阅（subscribe） 基于标准的ROS2 消息 sensor_msgs/Image 的交互
接口说明	实时获取右手腕部相机的彩色 RGB 图像
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息
	uint32 height：图像高度（行数）
	uint32 width：图像宽度（列数）
	string encoding：图像编码格式
	uint8 is_bigendian：字节序标识，0 表示小端序，1 表示大端序 ，通常取值为 0
	uint32 step：每行字节数，step = width * 每个像素字节数
	uint8[] data：图像数据本体（按行存储），数据长度 = height * step

Topic	/camera_r/aligned_depth_to_color/image_raw
生效版本	≥V0.3.0
接入方式	订阅（subscribe） 基于标准的ROS2 消息 sensor_msgs/Image 的交互
接口说明	实时获取右手腕部相机的深度图
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息
	uint32 height：图像高度（行数）
	uint32 width：图像宽度（列数）
	string encoding：图像编码格式
	uint8 is_bigendian：字节序标识，0 表示小端序，1 表示大端序 ，通常取值为 0

	uint32 step：每行字节数，step = width * 每个像素字节数
	uint8[] data：图像数据本体（按行存储），数据长度 = height * step

Topic	/camera_l/aligned_depth_to_color/camera_info
生效版本	≥V0.3
接入方式	订阅（subscribe） 使用ROS2的标准sensor_msgs/CameraInfo消息格式
接口说明	获取左手腕部相机的相机内参、畸变信息
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id="chassis_camera_rear_link" (rear)

Topic	/camera_r/aligned_depth_to_color/camera_info
生效版本	≥V0.3
接入方式	订阅（subscribe） 使用ROS2的标准sensor_msgs/CameraInfo消息格式
接口说明	获取右手腕部相机的相机内参、畸变信息
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id="chassis_camera_rear_link" (rear)

IMU传感器（惯性测量单元）

使用ROS2标准的 sensor_msgs/Imu.msg 消息类型

Topic	/imu/chassis/refined
生效版本	≥V0.2
接入方式	订阅（subscribe） 基于标准的ROS2 消息 sensor_msgs/Imu.msg 的交互
接口说明	实时获取底盘 IMU 运动状态，包括姿态（四元数）、角速度、线加速度及协方差
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id = "imu_chassis_link"

消息类型：std_msgs/Header
orientation：底盘姿态（四元数 $[x, y, z, w]$ ，满足 $x^2 + y^2 + z^2 + w^2 = 1$ ），描述俯仰、横滚、偏航角
消息类型：geometry_msgs/Quaternion
float64[9] orientation_covariance：姿态协方差，3x3 协方差矩阵，反映姿态测量的不确定性
angular_velocity：底盘绕 x/y/z 轴的旋转角速度，单位是弧度/秒（rad/s）
消息类型：geometry_msgs/Vector3
float64[9] angular_velocity_covariance：角速度协方差，3x3 协方差矩阵，反映角速度测量的不确定性
linear_acceleration：底盘沿 x/y/z 轴的线加速度，单位是 m/s^2
消息类型：geometry_msgs/Vector3
float64[9] linear_acceleration_covariance：加速度协方差，3x3 协方差矩阵，反映线加速度测量的不确定性

激光雷达

使用标准的ROS2 消息 sensor_msgs/LaserScan.msg格式

Topic	/laser_scan/front
生效版本	≥V0.2
接入方式	订阅（subscribe）
接口说明	实时获取前激光雷达的扫描数据，包含角度、时间、距离、强度等信息
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id = "laser_forward_link" 消息类型：std_msgs/Header
	float32 angle_min：起始角度，单位为弧度（rad）
	float32 angle_max：结束角度，单位为弧度（rad）
	float32 angle_increment：相邻两个点之间的角度间隔，单位为弧度（rad）
	float32 time_increment：相邻两个点的采样时间间隔，单位为 s
	float32 scan_time：扫描一圈花费的总时间，单位为 s

float32 range_min：最小有效测距值，单位为 m
float32 range_max：最大有效测距值，单位为 m
float32[] ranges：距离数据，数组长度 = 扫描点总数（由 angle_min、angle_max、angle_increment 计算），单位为 m
float32[] intensities：激光反射强度数据，与 ranges 数组长度一致，反映目标反射率差异

Topic	/laser_scan/rear
生效版本	≥V0.2
接入方式	订阅（subscribe）
接口说明	实时获取后激光雷达的点云
接口参数	header：标准消息头，包含时间戳（stamp）、坐标系（frame_id）等元信息， header.frame_id = "laser_back_link" 消息类型：std_msgs/Header
	float32 angle_min：起始角度，单位为弧度（rad）
	float32 angle_max：结束角度，单位为弧度（rad）
	float32 angle_increment：相邻两个点之间的角度间隔，单位为弧度（rad）
	float32 time_increment：相邻两个点的采样时间间隔，单位为 s
	float32 scan_time：扫描一圈花费的总时间，单位为 s
	float32 range_min：最小有效测距值，单位为 m
	float32 range_max：最大有效测距值，单位为 m
	float32[] ranges：距离数据，数组长度 = 扫描点总数（由 angle_min、angle_max、angle_increment 计算），单位为 m
	float32[] intensities：激光反射强度数据，与 ranges 数组长度一致，反映目标反射率差异

里程计数据的发布接口

使用标准ROS2 消息 nav_msgs::msg::Odometry

Topic	/feedback/odom
生效版本	≥V0.2

接入方式	发布（publish） 标准ROS2消息nav_msgs/Odometry格式
接口说明	输出机器人里程计数据，包括位姿、运动速度及坐标系信息
接口参数	header，包含时间戳（stamp）、坐标系（frame_id）等元信息， frame_id = "odom"（里程计坐标系，关联地图与机器人运动） float64[] pose.pose.position：笛卡尔坐标系下的位置矩阵，格式为[x,y,z]，单位是 m float64[] pose.pose.orientation：笛卡尔坐标系下的姿态（四元数），格式为[x,y,z,w] child_frame_id = "base_footprint"（机器人底盘中心坐标系，关联里程计与本体运动） twist { //运动速度 float64[] linear：线速度，格式为[linear.x, linear.y, linear.z]，单位是m/s，取值范围是 [-1.0, 1.0] float64[] angular：角速度，格式为[angular.x, angular.y, angular.z]，单位是弧度/秒（rad/s），取值范围是[-1.0, 1.0] }

6. 系统状态与诊断接口

Topic	/supervisor/system_status
生效版本	≥V0.2
接入方式	发布（publish）
接口说明	系统监控节点（supervisor）实时发布机器人系统的当前运行状态，包括当前模式和开机启动模式 使用自定义的 supervisor_interfaces/msg/SystemStatus 消息格式
接口参数	current_mode：当前系统模式 使用自定义的 supervisor_interfaces/msg/SystemMode 消息格式 uint8 mode{ //模式值，枚举 0; //未定义 1; //自动 2; //遥控 3; //建图 4; // ACT

<pre>99; // 待命，所有服务停止 } string mode_name{ //模式名称 "UNDEFINED"; //未定义 "TELEOPERATION"; //遥操作模式 "MAPPING"; //建图模式 "AUTONOMOUS"; //自动模式 "ACT"; //特定操作模式 "STANDBY"; //待命模式，所有服务停止 }</pre>	
startup_mode：系统开机启动时的初始状态 使用自定义的 supervisor_interfaces/msg/SystemMode 消息格式 <pre>uint8 mode{ //模式值，枚举 0; //未定义 1; //自动 2; //遥操 3; //建图 4; // ACT 99; // 待命，所有服务停止 } string mode_name{ //模式名称 "UNDEFINED"; //未定义 "TELEOPERATION"; //遥操作模式 "MAPPING"; //建图模式 "AUTONOMOUS"; //自动模式 "ACT"; //特定操作模式 "STANDBY"; //待命模式，所有服务停止 }</pre>	

Topic	/supervisor/change_system_mode
生效版本	≥V0.2
接入方式	Action

	使用自定义的 supervisor_interfaces/action/ChangeSystemMode 消息格式
接口说明	切换当前系统模式
接口参数	Goal：目标，客户端发送给服务器的请求，包含两个字段 target_mode{ //要切换的目标模式值 使用自定义的 supervisor_interfaces/msg/SystemMode 消息格式 1; //自动 2; //遥控 3; //建图 4; //ACT } bool startup：是否设置为开机自启，false 表示不开机自启，true表示开机自启 Result：服务器执行完成后返回给客户端的结果，包含两个字段 bool success：模式切换是否成功，false表示模式未切换，true表示模式切换成功 string message：模式切换执行结果的描述信息，如“切换至自动模式成功” Feedback：服务器在执行过程中周期性地向客户端发送的进度反馈，包含两个字段 float32 progress：模式切换的进度，取值为 [0.0, 100.0] string status：当前执行切换的实时状态描述，如“正在关闭当前服务”

诊断信息各模块字段明细

系统内包含诊断信息得到topic名称为/diagnostics_agg。

(1) 本体关节 (manipulator)

key	类型	含义
<关节名>_position	FLOAT rad	关节位置
<关节名>_velocity	FLOAT rad/s	关节速度
<关节名>_effort	FLOAT 牛米	关节力矩
<关节名>_operational	BOOL	关节是否上电使能（OP）： True 已使能 / False 未使能
<关节名>_alert	INT 错误码	关节错误码
<关节名>_temperature	FLOAT °C	关节电机温度

(2) 底盘 (chassis)

运动状态

key	类型	含义
linear_x / linear_y / linear_z	FLOAT m/s	三向线速度
angular_x / angular_y / angular_z	FLOAT rad/s	三向角速度
position_x / position_y / position_z	FLOAT m	底盘位置
orientation_x / orientation_y / orientation_z / orientation_w	FLOAT	底盘朝向四元数

底盘电机

key	类型	含义
left_wheel_speed_rpm / right_wheel_speed_rpm	FLOAT rpm	左右轮转速
left_wheel_current_a / right_wheel_current_a	FLOAT A	左右轮电流
left_motor_temperature / right_motor_temperature	FLOAT °C	左右电机温度
chassis_driver_temperature	FLOAT °C	底盘驱动器温度
chassis_state	INT	底盘电机状态：0 未使能 / 1 使能 / 2 故障 / 3 断连 /
chassis_error_code_left / chassis_error_code_right	INT	左右电机错误码，0 无错误

(3) 末端执行器 (end_effector)

<side> ∈ { left, right }。

key	类型	含义
<side>_joint_count	INT	该侧灵巧手关节数
<side>_<关节名>_position	FLOAT %	关节闭合程度 0-100%
<side>_<关节名>_velocity	FLOAT %	关节运动速度 0-100%
<side>_<关节名>_state	INT	关节运行状态：0 空闲到位 / 1 运动中 / 2 堵转保护停止 / 3 电机报错 / 4 夹住物体 / 5 DROPPED 物体掉落 / 100 串口未连接 / 255 未知异常

(4) 传感器 (sensors)

传感器联通性

key	类型	含义
left_hand_camera_reachability	BOOL	左腕部相机联通
right_hand_camera_reachability	BOOL	右腕部相机联通
head_camera_reachability	BOOL	头部相机联通
front_chassis_camera_reachability	BOOL	底盘前相机联通
rear_chassis_camera_reachability	BOOL	底盘后相机联通
chassis_laser_reachability	BOOL	底盘激光雷达联通

安全输入

key	类型	含义
estop_state	BOOL	急停按钮是否触发

bumper_strip_front_state	BOOL	前防撞条是否触发
bumper_strip_rear_state	BOOL	后防撞条是否触发
charger_connect_state	BOOL	充电器是否接入
voice_wakeup_state	BOOL	语音唤醒是否触发

(5) 运维监控 (system_monitor)

资源占用 (PC1 / PC2)

key	类型	含义
pc1_cpu_usage / pc2_cpu_usage	FLOAT %	CPU 占用 0-100
pc1_ram_usage / pc2_ram_usage	FLOAT %	内存占用 0-100
pc1_disk_usage / pc2_disk_usage	FLOAT %	硬盘占用 0-100
pc1_gpu_usage / pc2_gpu_usage	FLOAT %	GPU 占用 0-100

电源板与电池

key	类型	含义
power_board_operational	BOOL	电源板是否在线且数据新鲜： true / false
battery_voltage	FLOAT	电池电压
battery_current	FLOAT	电池电流
battery_percentage	FLOAT %	电池剩余电量 0-100
power_supply_health	INT	电池健康状态码
battery_temperature	FLOAT	电池温度
battery_charging	BOOL	

		是否充电中： <code>true</code> / <code>false</code>
<code>voltage_warn_low</code> / <code>voltage_warn_high</code>	FLOAT	电压告警下限/上限
<code>peripheral_connectivity_bitmap</code>	INT	外设联通位图
<code>missing_required_channels</code>	STR	缺失的必选通道列表，无缺失为 <code>none</code>
<code>firmware_version</code>	STR	固件版本，未知为 <code>unknown</code>
<code>hardware_version</code>	STR	硬件版本，未知为 <code>unknown</code>
<code>sn_code</code>	STR	电源板 SN 码
<code>pc_sn_code</code>	STR	PC SN 码
<code>channel_<通道名></code>	BOOL	各电源通道是否使能： <code>true</code> / <code>false</code> （每个通道一条）

通道：

- `channel_V48_PW_BMC`
- `channel_V48_PW_ARML`
- `channel_V48_PW_ARMR`
- `channel_V12_OP1`
- `channel_V12_LIGHT`
- `channel_V12_MCB1`
- `channel_V12_SDA`
- `channel_V12_FLOOR`
- `channel_V12_DISPLAY`
- `channel_V12_SWITCH`
- `channel_V12_LIDAR1`
- `channel_V12_LIDAR2`
- `channel_V12_SCAN1`

- channel_V12_SCAN2
- channel_V24_PW_CHASSIS1
- channel_V24_PW_CHASSIS2
- channel_V5_LORA
- channel_V5_WIFI
- channel_V5_5G
- channel_V5_FVE
- channel_V5_BVE
- channel_V5_OP1
- channel_V5_OP2
- channel_V5_LDIF
- channel_V5_RDIF
- channel_V5_BLVE
- channel_V48_SOFT_START
- channel_CHARGEAU_CTL
- channel_PW_MAIN
- channel_V12PW_EN
- channel_V12PW_CTOP
- channel_V5_PW

(6) 网络

key	类型	含义
pc1_wlan_ip	STR	PC1 无线 IP，无则 unavailable
pc2_wlan_ip	STR	PC2 无线 IP，无则 unavailable