

FTArm B9 机械臂HTTP/WS API 文档

适用工作区： Dragon_b9_Right（右臂，真机验证版） / Dragon_b9_Left（左臂）
服务组件： b9_web_server（随工作区分发，C++ 实现，零外部依赖） **协议：** HTTP/1.1 · HTTPS · WebSocket **数据格式：** JSON（UTF-8） **鉴权：** 无（见 §10.2）

目录

- [FTArm B9 机械臂HTTP/WS API 文档](#)
 - [目录](#)
- [1. 概述](#)
 - [1.1 简介](#)
 - [1.2 服务地址](#)
 - [1.3 左右臂工作区字段对照](#)
 - [1.4 快速开始](#)
- [2. 通用约定](#)
 - [2.1 请求与响应规范](#)
 - [2.2 阻塞模型与超时矩阵](#)
 - [2.3 并发与抢占语义](#)
 - [2.4 坐标系与单位](#)
 - [2.5 默认运动参数](#)
 - [2.6 直线安全工作域](#)
- [3. HTTP REST 接口详解](#)
 - [3.1 get_status — 获取系统状态](#)
 - [3.2 get_pose — 获取末端位姿](#)
 - [3.3 get_motors — 获取电机诊断](#)
 - [3.4 get_controllers — 健康检查](#)
 - [3.5 move_end_effector — 末端运动（自由/直线/圆弧）★](#)
 - [3.6 move_joints — 关节运动](#)
 - [3.7 cancel — 取消运动标记](#)
 - [3.8 enable — 电机使能](#)
 - [3.9 disable — 电机失能（软急停）](#)
 - [3.10 set_control_mode — 切换控制模式](#)
 - [3.11 teach_record — 示教录制](#)
 - [3.12 set_teach_mode — 示教模式开关](#)
 - [3.13 playback — 轨迹回放](#)
 - [3.14 shutdown — 远程关机](#)
 - [3.15 get_panel — 控制面板页](#)

- 4. WebSocket 接口详解
 - 4.1 连接与生命周期
 - 4.2 指令消息总表
 - 4.3 推送帧结构
 - 4.4 stage 状态机
 - 4.5 完整交互时序示例
- 5. 数据字典
 - 5.1 PoseTarget 对象
 - 5.2 关节顺序与限位
 - 5.3 CtrlMode 枚举
 - 5.4 ErrorCode 枚举
 - 5.5 message 语义速查
- 6. 接口支持列表
- 7. 典型业务流程
 - 7.1 安全开机与首次运动
 - 7.2 直线运动流水线
 - 7.3 示教 → 保存 → 回放
 - 7.4 安全收尾
- 8. 客户端参考实现
 - 8.1 Python 同步 SDK (生产级封装)
 - 8.2 Python 异步 WebSocket (带进度)
 - 8.3 浏览器 JavaScript
 - 8.4 curl 全接口合集
- 9. 常见问题 FAQ
- 10. 附录
 - 10.1 服务端启动参数
 - 10.2 安全部署建议
 - 10.3 术语表

1. 概述

1.1 简介

B9 远程控制 API 提供了一套与 B9 机械臂控制系统通信的接口，通过 HTTP / WebSocket 协议 (JSON 报文) 实现控制与数据交互，服务以 IP:端口 作为唯一标识。核心能力：

能力	REST	WebSocket	说明
末端自由运动	✓	✓	任意可达位姿
末端直线运动	✓	✓	笛卡尔直线插值，真机验证

末端圆弧运动	✓	✓	指定/自动圆心
关节空间运动	✓	✓	7 关节目标角
只规划预览	✓	✓	plan_only
运动进度实时反馈	✗	✓	stage/progress 推送
状态高频推送	✗ (轮询)	✓ (20ms)	关节位置流
电机使能/失能/模式	✓	✓	
拖动示教 + 轨迹回放	✓	✓	一键进出示教
电机诊断	✓	—	错误码/反馈龄/使能位
远程关机	✓	✓	REST 需二次确认
网页控制面板	✓ GET /	—	免开发即用

1.2 服务地址

通道	默认地址	说明
HTTP	http://<主机IP>:8087	局域网推荐
HTTPS	https://<主机IP>:8443	自签证书，首次访问需信任
WebSocket	ws://<主机IP>:8087/ws	同端口 /ws 路径
控制面板	http://<主机IP>:8087/	浏览器直接打开

主机 IP：服务器终端执行 `hostname -I` 取第一个地址。端口修改见 §10.1。

1.3 左右臂工作区字段对照

两工作区 API 结构完全一致，仅命名不同（同一时刻仅运行其一）。本文以右臂为主体，左臂按下表替换：

项目	右臂 (Dragon_b9_Right)	左臂 (Dragon_b9_Left)
mode 取值	"right_arm"	"left_arm"
目标位姿对象键	"right"	"left"
关节数组键	"right_joints"	"left_joints"
状态帧关节键	right_joints	left_joints
使能/失能返回键	"right"	"left"
直线安全工作域(Y)	-0.28 ~ -0.04 m	+0.04 ~ +0.28 m

1.4 快速开始

```
B=http://192.168.1.100:8087 # ← 换成你的主机 IP

curl $B/api/status # ① 健康检查
curl -X POST $B/api/enable -d '{}' # ② 使能电机
curl -X POST $B/api/end_effector -H 'Content-Type: application/js
  "mode":"right_arm",
  "right":{"x":0.275,"y":-0.28,"z":0.48,"roll":-3.141,"pitch":-1.
  "cartesian_linear":true}' # ③ 直线运动
# → {"success":true,"message":"Cartesian execution finished for r
curl $B/api/pose # ④ 确认到位
```

2. 通用约定

2.1 请求与响应规范

- 请求头：Content-Type: application/json; POST 请求体为 JSON 对象，GET 无参数
- JSON 解析为轻量实现**：字段名精确匹配（区分大小写）、不支持注释与尾逗号、布尔用 true/false、目标位姿必须是**嵌套对象** ("right": {...})

HTTP 状态	含义
200	请求受理（业务成败看响应体 success）
400	运动类接口业务失败（同时 success:false）

通用响应体

字段	类型	说明
success	bool	业务是否成功
message	string	结果描述或失败原因（英文）

2.2 阻塞模型与超时矩阵

REST 运动类接口为**同步阻塞式**——HTTP 连接保持到动作完成才返回：

接口	阻塞行为	服务器侧上限	客户端建议 timeout
POST /api/end_effector	阻塞至运动 完成	60 s	≥ 90 s

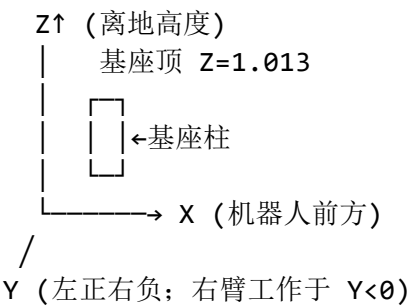
POST /api/joints	阻塞至运动完成	无显式上限	$\geq 120\text{ s}$
POST /api/playback	阻塞至回放完成	300 s	$\geq 310\text{ s}$
POST /api/teach	阻塞至服务响应	10 s	15 s
服务类 (enable 等)	短阻塞	2s 等待 + 5s 响应	10 s
GET 类	即时	—	5 s

低速长距离运动可能超过 60s → 提高 velocity_scaling 或改用 WebSocket (无 60s 限制且有进度)。

2.3 并发与抢占语义

场景	行为
运动中再发运动请求	新目标抢占旧目标 ：旧请求以失败结束，新目标立即执行
多客户端同时控制	服务器不做互斥，遵循抢占规则—— 请在业务层保证单一控制方 ，其余只读
回放中发运动	相互冲突，勿混用

2.4 坐标系与单位



量	单位
位置 x/y/z	m (world 系)
姿态 roll/pitch/yaw	rad (XYZ 固定轴欧拉角)
关节角	rad

速度/加速度缩放 无量纲 0.01–1.0（关节限速百分比）

2.5 默认运动参数

字段省略或传 0 时生效。出厂默认即**验证脚本同款安全参数**——不确定就全部省略：

参数	默认值	说明
velocity_scaling	0.12	12% 关节限速
acceleration_scaling	0.12	
cartesian_eef_step	0.025 m	直线插值步长
cartesian_min_fraction	0.85	直线成立最低 IK 成功率
planning_time	20 s	回退规划时限

2.6 直线安全工作域

固定推荐姿态 roll=-3.141, pitch=-1.552, yaw=3.141、X=0.275 平面（经 324 点可达性采样验证）：

工作区	Y	Z
右臂	-0.28 ~ -0.04	0.44 ~ 0.52
左臂	+0.04 ~ +0.28	0.44 ~ 0.52

越界后果：直线 IK 成功率不足 → **自动回退自由路径**（响应 message 含 OMPL 字样，路径不再是直线）或失败。

3. HTTP REST 接口详解

每个接口按固定八要素编写：接口定义表 → 请求参数表 → 返回参数表 → curl 示例 → Python 示例 → 返回示例 → 失败响应 → 注意事项（无则省略）。Python 示例默认已 import requests；B = "http://<IP>:8087"。

3.1 get_status — 获取系统状态

项	值
请求方式	GET

接口地址	/api/status
阻塞	否
描述	时间戳、运动中标记、关节位置快照。低频轮询建议 $\geq 200\text{ms}$ 间隔；高频监控用 WebSocket

返回参数

字段	类型	说明
timestamp	double	服务器时间（秒）
moving	bool	是否有运动进行中（软标记）
moveit_available	bool	恒 true
right_joints	object null	{ 关节名: 位置(rad) }; 未就绪为 null
right_pose	null	预留（末端位姿用 3.2）

```
curl $B/api/status
```

```
requests.get(f"{B}/api/status", timeout=5).json()
```

返回示例

```
{ "timestamp": 1784269396.71, "moving": false, "moveit_available": true,
  "right_joints": { "right_shoulder_pitch_joint": 0.5124, "right_shoul
    "right_shoulder_yaw_joint": -0.0021, "right_elbow_roll_joint": -0.8
    "right_elbow_yaw_joint": 0.0140, "right_wrist_pitch_joint": -0.6621
    "right_wrist_yaw_joint": 0.0093 }, "right_pose": null }
```

3.2 get_pose — 获取末端位姿

项	值
请求方式	GET
接口地址	/api/pose
阻塞	否
描述	末端连杆在 world 系的实时位姿（TF 解算）

返回参数

字段	类型	说明
arm	string	"right" / "left"
pose	object null	系统刚启动数秒内 TF 未就绪为 null，稍候重试
pose.x / y / z	double	m
pose.roll / pitch / yaw	double	rad

```
curl $B/api/pose
```

返回示例

```
{"arm": "right", "pose": {"x": 0.2750, "y": -0.1600, "z": 0.4800, "roll": -3.1409, "pitch": -1.5519, "yaw": 3.1408}}
```

3.3 get_motors — 获取电机诊断

项	值
请求方式	GET
接口地址	/api/motors
阻塞	否
描述	每关节全部硬件接口值——远程判断“电机是否健康/是否上力”的唯一途径

返回结构：{关节名：{接口：值}}，未就绪返回 {}

接口	说明	健康判据
position / velocity / effort	实时位置(rad)/速度(rad/s)/力矩(Nm)	—
motor_error	电机错误码	0 正常；≥0x08 故障
fault	故障锁存	必须为 0；为 1 需断电复位
has_feedback	收到过反馈	1

feedback_age	距上次反馈秒数	< 0.1
enabled	使能状态	运动前必须为 1

```
curl $B/api/motors | python3 -m json.tool
```

```
m = requests.get(f"{B}/api/motors", timeout=5).json()
healthy = bool(m) and all(j["fault"]==0 and j["has_feedback"]==1
                           for j in m.values())
enabled = all(j["enabled"]==1 for j in m.values())
```

3.4 get_controllers — 健康检查

项	值
请求方式	GET
接口地址	/api/controllers
描述	轻量健康检查

返回示例

```
{"joint_state_available":true,"active":false}
```

3.5 move_end_effector — 末端运动（自由/直线/圆弧）

★

项	值
请求方式	POST
接口地址	/api/end_effector
阻塞	是，直至运动完成，服务器上限 60s
描述	驱动末端到目标位姿；三种路径：自由（默认）/ 直线 / 圆弧

请求参数

参数	类型	必填	默认	说明
mode	string	否	right_arm	本工作区臂名

right	object	是	—	目标位姿对象 (§5.1; 左臂工作区键名 left)
right.x / y / z	double	是	—	位置 (m)
right.roll / pitch / yaw	double	是 *	—	姿态 (rad); position_only=true 时忽略
cartesian_linear	bool	否	false	true = 末端走直线
cartesian_arc	bool	否	false	true = 圆弧 (与直线互斥)
arc_center_x/y/z	double	否	0	圆弧圆心; 全 0 自动计算
velocity_scaling	double	否	0.12	0.01–1.0; 真机建议 ≤0.3
acceleration_scaling	double	否	0.12	
cartesian_eef_step	double	否	0.025	直线步长 (0, 0.1] m
cartesian_min_fraction	double	否	0.85	[0,1]; 低于则回退自由规划
planning_time	double	否	20	回退规划时限 s
position_only	bool	否	false	只约束位置
plan_only	bool	否	false	只规划不执行 (安全预览)

返回参数

字段	类型	说明
success	bool	
message	string	路径语义判据见 §5.5 (含 OMPL 即未走直线)

直线 (默认安全参数)

```
curl -X POST $B/api/end_effector -H 'Content-Type: application/json' \
  -d '{"mode":"right_arm",
    "right":{"x":0.275,"y":-0.28,"z":0.48,"roll":-3.141,"pitch":-1.
    "cartesian_linear":true}'
```

先预览后执行

```
curl ... -d '{"...", "cartesian_linear":true, "plan_only":true}'
curl ... -d '{"...", "cartesian_linear":true}'
```

高速高精度直线

```
curl ... -d '{"...", "cartesian_linear":true, "velocity_scaling":0.

```

```
def linear(x, y, z, vel=0.12):
    r = requests.post(f"{B}/api/end_effector", json={
        "mode": "right_arm",
        "right": {"x":x, "y":y, "z":z, "roll":-3.141, "pitch":-1.
        "cartesian_linear": True, "velocity_scaling": vel}, timeo
    j = r.json()
    assert j["success"], j["message"]
    assert "OMPL" not in j["message"], "未走直线(已回退): " + j["me
    return j
```

返回示例

```
{"success":true,"message":"Cartesian execution finished for right
```

失败响应

```
{"success":false,"message":"Invalid mode or missing target"}
{"success":false,"message":"All planning strategies failed for ri
{"success":false,"message":"Motion timeout (60s)"}
{"success":false,"message":"EE action server not available"}
```

注意事项

- 1. 直线目标保持在 §2.6 工作域内；message 含 OMPL 即偏离直线，应视为业务告警
- 2. 60s 上限：低速长距离改 WebSocket 或提速
- 3. 新请求抢占旧运动 (§2.3)

3.6 move_joints — 关节运动

项	值
请求方式	POST
接口地址	/api/joints
阻塞	是，直至完成
描述	7 关节目标角，关节空间规划执行

请求参数

参数	类型	必填	默认	说明
----	----	----	----	----

mode	string	否	right_arm
right_joints	double[7]	是	— 顺序与限位见 §5.2（左臂键名 left_joints）
velocity_scaling	double	否	0.3
acceleration_scaling	double	否	0.1
plan_only	bool	否	false

```
curl -X POST $B/api/joints -H 'Content-Type: application/json' -d
  "mode":"right_arm",
  "right_joints":[0.0, 0.5, 0.0, -1.0, -0.1, -1.0, 0.0],
  "velocity_scaling":0.2}'
```

返回/失败响应

```
{ "success":true, "message":"Joint motion executed" }
{ "success":false, "message":"Invalid joint array" } // 数组缺失/长
{ "success":false, "message":"Planning failed" } // 目标超限/自
```

3.7 cancel — 取消运动标记

项	值
请求方式	POST
接口地址	/api/cancel
描述	复位"运动中"软标记

⚠ 局限 **不会中断已在执行的轨迹。**真停 = 新目标抢占（3.5）或失能（3.9）

```
curl -X POST $B/api/cancel
# { "success":true, "message":"Cancel requested" }
```

3.8 enable — 电机使能

项	值
请求方式	POST
接口地址	/api/enable
请求体	{}

描述	电机上力锁住当前姿态
----	------------

返回参数（键名随工作区：右臂 right / 左臂 left）

字段	类型	说明
right.success	bool	
right.message	string	成功描述 / unavailable 硬件服务未就绪 / timeout 5s 无响应

```
curl -X POST $B/api/enable -d '{}'  
# {"right":{"success":true,"message":"7 motors enabled"}}
```

注意：若服务器主栈以只读安全模式部署（auto_enable=false），使能后运动接口仍不会驱动实机（返回成功但机械臂不动）。属服务器部署配置，请联系管理员以正常模式重启主栈。

3.9 disable — 电机失能（软急停）

项	值
请求方式	POST
接口地址	/api/disable
描述	立即切断全部电机力矩——远控通道下最快的软急停

 **警告** **失能瞬间手臂因重力下坠。** 确保下方无人/物，或先运动至低位

```
curl -X POST $B/api/disable -d '{}'  
# {"right":{"success":true,"message":"7 motors disabled"}}
```

3.10 set_control_mode — 切换控制模式

项	值
请求方式	POST
接口地址	/api/control_mode
描述	切换电机底层模式（\$5.3）。示教模式由 3.12 自动管理，一般无需手调

请求参数

参数	类型	必填	默认	可选值
----	----	----	----	-----

mode	string	否	pos_vel	pos_vel / mit / pos_vel_csp
------	--------	---	---------	-----------------------------

```
curl -X POST $B/api/control_mode -d '{"mode":"pos_vel"}'  
# {"right":{"success":true,"message":"mode set to pos_vel"}}
```

3.11 teach_record — 示教录制

项	值
请求方式	POST
接口地址	/api/teach
描述	控制录制流水（须已处于示教模式，见 3.12）

请求参数

参数	类型	必填	默认	说明
command	string	是	start	start / stop / save / cancel
filename	string	save 必填	—	相对名自动存服务器 ~/trajectories/; 支持 .csv .yaml; 可绝对路径
gravity_comp	bool	否	true	录制时重力补偿托举

```
curl -X POST $B/api/teach -d '{"command":"save","filename":"demo."
```

返回示例

```
{ "success":true,"message":"Trajectory saved","trajectory_id":"/ho
```

trajectory_id 可直接作为 3.13 回放入参。

3.12 set_teach_mode — 示教模式开关

项	值
请求方式	POST
接口地址	/api/teach_mode

描述	一键进/出示教。进入自动：停轨迹控制器→电机切零力矩→启示教控制器→开始录制；退出反向恢复
----	---

请求参数

参数	类型	必填	默认
enable	bool	否	true

```
curl -X POST $B/api/teach_mode -d '{"enable":true}' # 进入, 手
curl -X POST $B/api/teach_mode -d '{"enable":false}' # 退出, 恢
# {"success":true}
```

注意：示教模式期间 3.5/3.6 运动接口不可用；退出后自动恢复。

3.13 playback — 轨迹回放

项	值
请求方式	POST
接口地址	/api/playback
阻塞	是，上限 300s（客户端 timeout ≥310s）
描述	回放示教轨迹（自动平滑/抽稀/自碰撞检查）

请求参数

参数	类型	必填	默认	说明
trajectory_id	string	是	—	文件名（自动拼默认目录）或绝对路径
speed_scale	double	否	1.0	速度倍率 >0
loop_count	int	否	1	循环次数

```
curl -X POST $B/api/playback --max-time 310 \
-d '{"trajectory_id":"demo.csv","speed_scale":0.8,"loop_count":
```

返回/失败响应

```
{"success":true,"message":"Playback complete"}
{"success":false,"message":"Playback timeout (300s)"}
{"success":false,"message":"Failed to load: /home/xia17/trajector
{"success":false,"message":"183/200 waypoints in self-collision (
```

3.14 shutdown — 远程关机

项	值
请求方式	POST
接口地址	/api/shutdown
描述	关闭服务器主机（等效 shutdown -h now），二次确认防误触

请求参数

参数	类型	必填	说明
confirm	bool	是	必须 true，否则不执行

```
curl -X POST $B/api/shutdown -d '{"confirm":true}'  
# {"success":true,"message":"Shutting down..."}
```

3.15 get_panel — 控制面板页

项	值
请求方式	GET
接口地址	/
描述	内置网页控制面板（单文件 HTML，原生 JS）。状态监控、末端/关节控制（含直线勾选）、示教、回放、使能失能、关机

4. WebSocket 接口详解

4.1 连接与生命周期

```
connect ws://<主机IP>:8087/ws  
| 连接即开始接收 20ms 周期 status 帧（无需订阅）  
|— send {"type":..., "data":{...}} ← 随时发指令  
|— recv {"type":"status"|"feedback"|"result"|"<回执>,...}  
|— close / 断线 → 客户端自行重连（建议 3s 退避）
```

- 状态帧对**所有连接广播**；指令回执与 feedback/result 仅发给发起连接

- 服务器不主动断开空闲连接；推送间隔可由启动参数 `web_ws_interval_ms` 调整
- 断线重连后先 `GET /api/status` 对齐状态，勿重发进行中的目标

4.2 指令消息总表

统一包装: `{"type": "<类型>", "data": {...}}`, `data` 与同名 REST 请求体一致:

type	等价 REST	差异说明
end_effector	3.5	异步 : 即时受理, 过程推 <code>feedback</code> , 完成推 <code>result</code> , 无 60s 限制
joints	3.6	异步, 完成推 <code>result</code>
cancel	3.7	无回执
enable / disable	3.8 / 3.9	回执 = REST 响应体
control_mode	3.10	同上
teach	3.11	回执含 <code>trajectory_id</code>
teach_mode	3.12	回执 <code>{"success": ...}</code>
playback	3.13	即时回 <code>Playback started</code> , 完成另推 <code>result</code>
pose	3.2	回执 <code>{"pose": {...}}</code>
shutdown	3.14	⚠ 无需 confirm, 发送即关机 ——客户端必须自行二次确认

4.3 推送帧结构

① StatusFrame (20ms 周期)

```
{"type": "status", "data": {"timestamp": 1784269396.71, "moving": true,
"moveit_available": true, "right_joints": {"right_shoulder_pitch_jo
"right_pose": null}}}
```

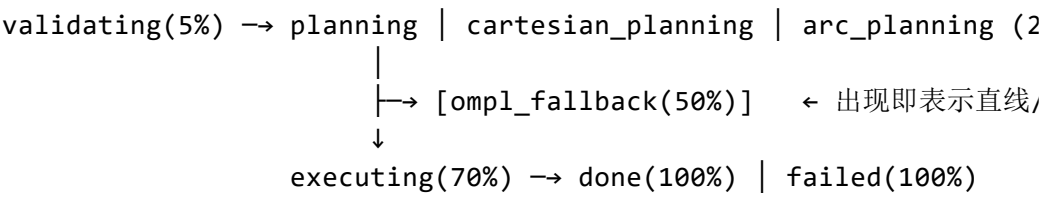
② FeedbackFrame (end_effector 目标专属)

```
{"type": "feedback", "data": {"stage": "cartesian_planning", "progress
"message": "Building linear trajectory for right_arm"}}
```

③ ResultFrame

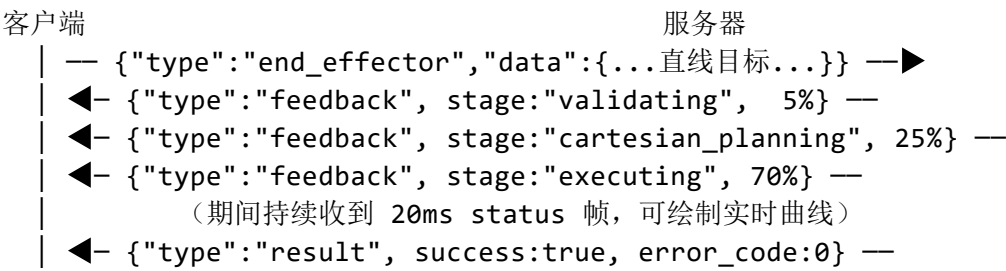
```
{
  "type": "result",
  "data": {
    "success": true,
    "error_code": 0,
    "message": "Cartesian execution finished for right_arm"
  }
}
```

4.4 stage 状态机



(`playback` 另有: `smoothing` → `ready` → `executing` → `resetting` → `done`)

4.5 完整交互时序示例



5. 数据字典

5.1 PoseTarget 对象

字段	类型	单位	必填
x, y, z	double	m	是
roll, pitch, yaw	double	rad	position_only 时可省

5.2 关节顺序与限位

索引	关节名	右臂限位(rad)	左臂限位(rad)
0	shoulder_pitch	±2.0	±2.0

1	shoulder_roll	-0.2 ~ 2.0	-0.2 ~ 2.0
2	shoulder_yaw	±2.0	±1.2
3	elbow_roll	±1.2	±1.2
4	elbow_yaw	±2.0	±2.0
5	wrist_pitch	±1.1	±1.1
6	wrist_yaw	±1.1	±1.1

全零位 = 手臂自然下垂。

5.3 CtrlMode 枚举

值	含义	用途
pos_vel	位置+速度限幅	常规运动（默认）
mit	力矩/阻抗模式	拖动示教
pos_vel_csp	周期同步位置	高频轨迹流

5.4 ErrorCode 枚举

码	名称	说明	处置
0	NONE	成功	—
1	INVALID_GOAL	字段缺失 / NaN / 参数越界	对照 3.5 参数表
2	PLANNING_FAILED	规划失败	目标不可达，查 §2.6
3	EXECUTION_FAILED	执行失败	查 3.3 电机诊断
4	CANCELED	被取消/抢占	预期行为
5	IK_FAILED	逆解失败	同 2
6	EXCEPTION	内部异常	取服务器日志反馈

5.5 message 语义速查

message 关键词	含义	处置
Cartesian execution finished	✅ 直线执行成功	—
Arc execution finished	✅ 圆弧执行成功	—

Execution finished (exact pose)	自由路径执行成功	—
OMPL execution finished	⚠ 已回退自由路径 (非直线)	目标回到工作域
Planning succeeded ...	plan_only 预览成功 (未运动)	—
All planning strategies failed	✗ 目标不可达	查 §2.6
Motion timeout (60s)	超 60s	提速或用 WS
Invalid mode or missing target	缺 right 对象	用嵌套对象格式
Invalid joint array	关节数组缺/长度≠7	修正数组
unavailable / timeout	底层服务不可达	检查主栈/硬件
EE action server not available	主栈未启动	启动主栈
Teach service not available	示教控制器未加载	用全家桶模式启动
Playback server not available	回放器未启动	同上

6. 接口支持列表

interface	REST	WebSocket	右臂工作区	左臂工作区
get_status	√	— (status 推送替代)	√	√
get_pose	√	√ (pose)	√	√
get_motors	√		√	√
get_controllers	√		√	√
move_end_effector (自由/直线/圆弧)	√	√ (end_effector)	√	√
move_joints	√	√ (joints)	√	√
cancel	√	√	√	√
enable / disable	√	√	√	√

set_control_mode	√	√ (control_mode)	√	√
teach_record	√	√ (teach)	√	√
set_teach_mode	√	√ (teach_mode)	√	√
playback	√	√	√	√
shutdown	√	√	√	√
运动进度反馈 (feedback)		√	√	√
20ms 状态推送 (status)		√	√	√
控制面板页 (GET /)	√		√	√

7. 典型业务流程

7.1 安全开机与首次运动

① GET /api/status	确认服务可达
② GET /api/motors	确认 has_feedback=1, fault=0
③ POST /api/enable	上力（确认 success）
④ GET /api/motors	确认 enabled 全为 1
⑤ POST /api/end_effector	plan_only:true 预览首目标
⑥ POST /api/end_effector	正式执行（低速 velocity_scaling:0.05）
⑦ GET /api/pose	校验到位

7.2 直线运动流水线

```
points = [(-0.16,0.48), (-0.28,0.48), (-0.28,0.52), (-0.04,0.52),
for y, z in points:
    r = linear(0.275, y, z, vel=0.15)           # 见 3.5 Python 示
    if "OMPL" in r["message"]:
        raise RuntimeError(f"({y},{z}) 未走直线, 检查工作域")
```

要点：串行等待每条响应（阻塞式天然保序）；勿并发连发。

7.3 示教 → 保存 → 回放

- ① POST /api/teach_mode {"enable":true} 进入示教（自动开始录制）
- ② 操作员拖动手臂演示
- ③ POST /api/teach {"command":"stop"}
- ④ POST /api/teach {"command":"save","filename":"task_A.csv"}
- ⑤ POST /api/teach_mode {"enable":false} 退出（恢复位置控制）
- ⑥ POST /api/playback {"trajectory_id":"task_A.csv"}

7.4 安全收尾

- ① POST /api/end_effector 直线降至低位（z:0.44, velocity_scaling:(
- ② 现场人员托住手臂
- ③ POST /api/disable 失能（下坠已受控）
- ④ （可选）POST /api/shutdown {"confirm":true}

8. 客户端参考实现

8.1 Python 同步 SDK（生产级封装）

```

"""b9_client.py - B9 机械臂 HTTP 客户端参考实现"""
import requests

class B9Error(RuntimeError): ...
class B9NotLinear(B9Error): ...           # 直线回退告警

class B9Client:
    RPY = (-3.141, -1.552, 3.141)         # 已验证推荐姿态

    def __init__(self, host, arm="right", port=8087, timeout=90):
        self.base, self.arm, self.timeout = f"http://{host}:{port}"

    def _get(self, path, t=5):
        return requests.get(self.base + path, timeout=t).json()

    def _post(self, path, body=None, t=None):
        return requests.post(self.base + path, json=body or {},
                             timeout=t or self.timeout).json()

    # ----- 查询 -----
    def status(self): return self._get("/api/status")
    def pose(self): return self._get("/api/pose")
    def motors(self): return self._get("/api/motors")

    def healthy(self):
        m = self.motors()
        return bool(m) and all(j["fault"] == 0 and j["has_feedback"]
                                for j in m.values())

    # ----- 电机 -----
    def enable(self):
        r = self._post("/api/enable", t=15)[self.arm]
        if not r["success"]: raise B9Error(f"使能失败: {r['message']}")
        return r

    def disable(self):
        # 远程软急停(注意掉臂!)
        return self._post("/api/disable", t=15)[self.arm]

    # ----- 运动 -----
    def move(self, x, y, z, rpy=None, *, linear=False, arc=False,
             vel=0.0, acc=0.0, step=0.0, plan_only=False, strict_
             rpy = rpy or self.RPY
        r = self._post("/api/end_effector", {
            "mode": f"{self.arm}_arm",
            self.arm: {"x": x, "y": y, "z": z,
                       "roll": rpy[0], "pitch": rpy[1], "yaw": rpy[2],
                       "cartesian_linear": linear, "cartesian_arc": arc,
                       "velocity_scaling": vel, "acceleration_scaling": acc,
                       "cartesian_eef_step": step, "plan_only": plan_only})
        if not r["success"]:
            raise B9Error(r["message"])
        if linear and strict_linear and "OMPL" in r["message"]:

```

```

        raise B9NotLinear(r["message"])
    return r

def line_to(self, x, y, z, **kw):    # 语义化直线
    return self.move(x, y, z, linear=True, **kw)

def joints(self, q7, vel=0.2):
    r = self._post("/api/joints", {"mode": f"{self.arm}_arm",
                                   f"{self.arm}_joints": list(q7),
                                   "velocity_scaling": vel}, t=180)
    if not r["success"]: raise B9Error(r["message"])
    return r

# ----- 示教/回放 -----
def teach(self, on=True):
    return self._post("/api/teach_mode", {"enable": on}, t=20)

def teach_save(self, name):
    return self._post("/api/teach", {"command": "save",
                                     "filename": name}, t=15)

def playback(self, traj, speed=1.0, loops=1):
    return self._post("/api/playback", {"trajectory_id": traj
                                         "speed_scale": speed, "loop_count": loc

if __name__ == "__main__":
    b9 = B9Client("192.168.1.100", arm="right")
    assert b9.healthy(), "电机状态异常"
    b9.enable()
    b9.line_to(0.275, -0.16, 0.48)    # 中心
    b9.line_to(0.275, -0.28, 0.52, vel=0.2)    # 对角直线
    print(b9.pose())

```

8.2 Python 异步 WebSocket (带进度)

```

import asyncio, json, websockets

async def line_to(host, x, y, z, on_progress=None):
    async with websockets.connect(f"ws://{host}:8087/ws") as ws:
        await ws.send(json.dumps({"type": "end_effector", "data":
            "mode": "right_arm",
            "right": {"x": x, "y": y, "z": z,
                "roll": -3.141, "pitch": -1.552, "yaw": 3.1
            "cartesian_linear": True}}))
        async for raw in ws:
            m = json.loads(raw)
            if m["type"] == "feedback" and on_progress:
                on_progress(m["data"]["progress"], m["data"]["sta
            elif m["type"] == "result":
                d = m["data"]
                if not d["success"]:
                    raise RuntimeError(f"[{d['error_code']}] {d['
                return d

asyncio.run(line_to("192.168.1.100", 0.275, -0.28, 0.48,
    on_progress=lambda p, s: print(f"{p*100:5.1f}

```

8.3 浏览器 JavaScript

```

const B = "http://192.168.1.100:8087";

async function lineTo(x, y, z) {
  const r = await fetch(`${B}/api/end_effector`, {
    method: "POST",
    headers: {"Content-Type": "application/json"},
    body: JSON.stringify({
      mode: "right_arm",
      right: {x, y, z, roll: -3.141, pitch: -1.552, yaw: 3.141},
      cartesian_linear: true }));
  const j = await r.json();
  if (!j.success) throw new Error(j.message);
  if (j.message.includes("OMPL")) console.warn("未走直线:", j.message);
  return j;
}

// 实时状态流
const ws = new WebSocket("ws://192.168.1.100:8087/ws");
ws.onmessage = (e) => {
  const m = JSON.parse(e.data);
  if (m.type === "status") updateDashboard(m.data.right_joints);
  if (m.type === "feedback") setProgress(m.data.progress);
  if (m.type === "result") onDone(m.data);
};

```

8.4 curl 全接口合集

```

B=http://192.168.1.100:8087
curl $B/api/status; curl $B/api/pose; curl $B/api/motors; curl
curl -X POST $B/api/enable -d '{}'
curl -X POST $B/api/disable -d '{}'
curl -X POST $B/api/control_mode -d '{"mode":"pos_vel"}'
curl -X POST $B/api/end_effector -H 'Content-Type: application/js
 '{"mode":"right_arm","right":{"x":0.275,"y":-0.28,"z":0.48,"roll
curl -X POST $B/api/joints -H 'Content-Type: application/json' -d
 '{"mode":"right_arm","right_joints":[0,0.5,0,-1.0,-0.1,-1.0,0],"
curl -X POST $B/api/teach_mode -d '{"enable":true}'
curl -X POST $B/api/teach -d '{"command":"save","filename":"demo.
curl -X POST $B/api/teach_mode -d '{"enable":false}'
curl -X POST $B/api/playback --max-time 310 -d '{"trajectory_id":
curl -X POST $B/api/cancel
curl -X POST $B/api/shutdown -d '{"confirm":true}'

```

9. 常见问题 FAQ

- Q1：运动接口返回 success 但机械臂没动？** A：① 服务器主栈以只读安全模式部署（联系管理员以正常模式重启）；② 电机未使能（查 3.3 的 enabled）；③ 目标即当前位置。
- Q2：怎么确认这一步真的走了直线？** A：message 为 Cartesian execution finished 即直线；出现 OMPL 即已回退非直线。建议客户端做成断言（见 8.1 B9NotLinear）。
- Q3：直线经常回退 OMPL？** A：目标超出工作域（§2.6）或姿态偏离推荐 RPY。先 plan_only:true 试探；必要时减小步长或放宽 cartesian_min_fraction。
- Q4：HTTP 请求 60 秒超时？** A：提高 velocity_scaling，或改用 WebSocket（异步无此限制）。
- Q5：能同时控制左臂和右臂吗？** A：不能。左右臂是互斥工作区（同一时间仅运行其一），单个服务实例只控制其对应臂。
- Q6：如何做急停按钮？** A：远控通道用 POST /api/disable（注意掉臂）；物理断电始终优先于任何软件手段。
- Q7：/api/cancel 为何停不下来？** A：它只复位软标记（3.7）。真正中断 = 新目标抢占或 disable。
- Q8：WebSocket 的 shutdown 没有 confirm？** A：是，发送即关机（4.2 表注）。客户端 UI 必须自行二次确认。
- Q9：多客户端会打架吗？** A：会。服务器不做互斥，后发目标抢占先发。业务层保证同一时刻单一控制方，其余只读。
- Q10：状态推送太频繁占带宽？** A：服务端调 web_ws_interval_ms 启动参数，或客户端自行降采样。

10. 附录

10.1 服务端启动参数

参数	默认	说明
web_http_port	8087	HTTP 端口
web_https_port	8443	HTTPS 端口
web_cert_dir	/tmp/b9_certs	自签证书目录（启动自动生成）
web_ws_interval_ms	20	状态帧推送间隔

10.2 安全部署建议

- 1. 无鉴权设计——**仅部署于受信隔离网段**；对外必须加反向代理（Nginx + BasicAuth/Token）与防火墙白名单
- 2. shutdown 与 disable 属高危接口，代理层建议单独鉴权
- 3. WSL 部署跨网访问需宿主机端口转发：netsh interface portproxy add v4tov4 listenport=8087 connectaddress=<WSL_IP> connectport=8087
- 4. 生产环境以正式证书替换自签证书

10.3 术语表

术语	释义
直线运动 (cartesian_linear)	末端在笛卡尔空间沿直线插值运动
回退 (OMPL fallback)	直线不成立时自动改用自由路径规划
使能 / 失能	电机上力 / 断力
示教	零力矩下人工拖动记录轨迹
工作域	指定姿态下经采样验证的直线可行区域
plan_only	只规划验证可行性、不产生实际运动

文档结束。接口行为以工作区源码 `b9_web_server/src/b9_web_server.cpp` 为最终依据；发现不一致请反馈修订。