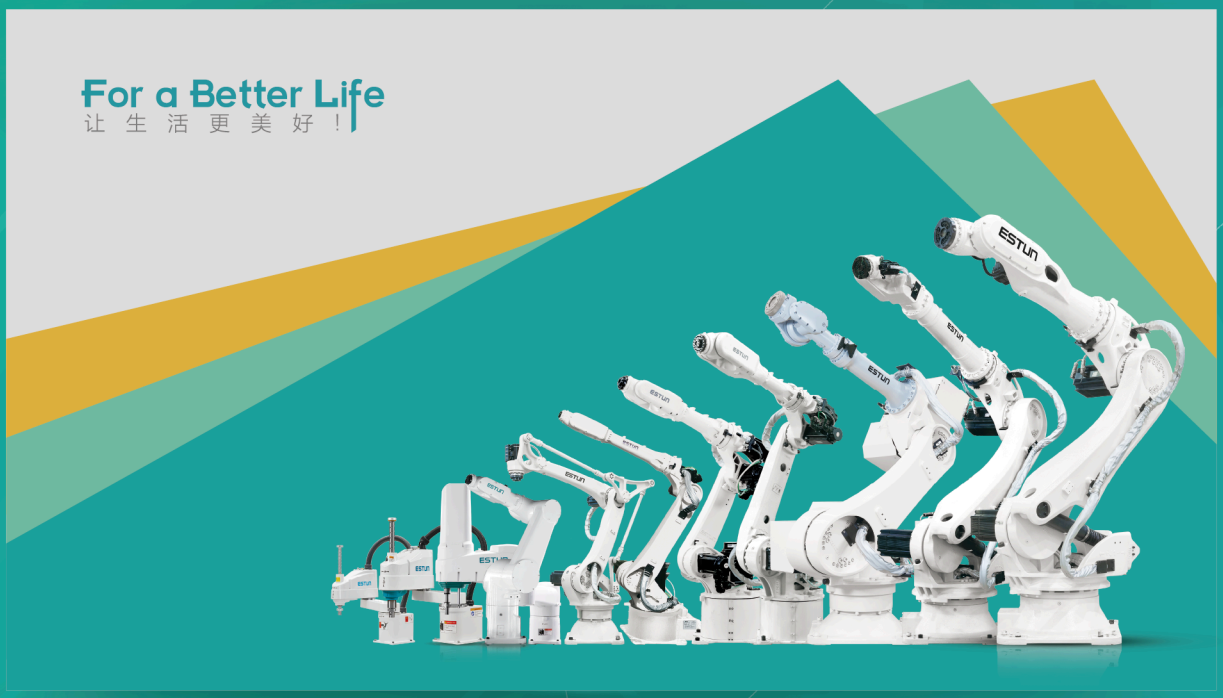
ESTUN
AUTOMATION

工业机器人
INDUSTRIAL
ROBOT



iER系列工业机器人

API使用手册

iEPM01V30406-01

自动化核心部件
运动控制系统

工业机器人及
智能系统工程

数字化产品
数字化服务

申明

版权与知识产权

本手册及其包含的所有文本、图像、设计、软件及相关信息（统称“文档材料”）均为南京埃斯顿自动化股份有限公司（简称“埃斯顿公司”）的独家财产，受中华人民共和国及相关国际版权法和条约保护。除非获得埃斯顿公司的明确书面授权，否则严禁对本文档材料进行任何形式的复制、分发、修改、公开展示或反向工程。

免责声明

1. 埃斯顿公司致力于确保手册信息的准确性，但不对其完整性、准确性或无误性作任何明示或默示的担保。埃斯顿公司将定期校对本手册的内容，并在之后的版本中作必要的更改，同时保留在不影响功能的情况下进行技术更改的权利，恕不另行通知。
2. 安全操作是用户的首要责任。用户有责任确保其人员已接受充分培训，并严格按照本手册、所有相关安全规程以及国家、地区和行业标准来安装、操作和维护本产品。任何因未遵守操作规程、擅自改装或使用非原装备件所导致的人身伤害、设备损坏或生产损失，埃斯顿公司概不负责。
3. 在任何情况下， 埃斯顿公司均不对因使用或无法使用本产品及手册而引发的任何间接损失、附带损失、利润损失、数据丢失或生产中断等后果性损害承担责任。

文档申明

1. 本手册为“Original Instruction”，经制造商埃斯顿公司核实确认。
2. 如需本手册的其他语言翻译版本，可向制造商或其授权代表申请获取。

前言

修订记录

序号	日期	版本	描述
1		V1.0	手册编写

更多资料

文档编号	文档名称
-	各机器人对应的使用手册。
-	各控制柜对应的安全手册。

资料获取

本文档不随产品发货，如需获取电子版PDF文件，可以通过以下方式获取： 登录埃斯顿公司官网网站（www.estun.com），“**服务与支持 -> 资料下载**”，搜索关键字并下载。

安全使用须知

本章说明为安全使用机器人而需要遵守的内容。在使用机器人之前，务必熟读并理解本章中所述内容。

使用埃斯顿机器人的公司、个人应该熟读所在地区、国家的标准和法律，并且安装适当的安全设施保护机器人的使用人员。使用前（安装、运转、保养、检修），请务必熟读并全部掌握本说明书和其他附属资料，在熟知全部设备知识、安全知识及注意事项后再开始使用。但是使用人员即使完全按照手册中给出的所有安全信息进行，埃斯顿公司也无法保证使用人员不会受到任何伤害。

使用人员的定义

1. 操作人员：
   - 执行机器人电源的开启（ON）与关闭（OFF）操作；
   - 通过示教器和控制柜操作面板启动、暂停或停止已预设的机器人运行程序；
   - 监控机器人日常运行状态，及时反馈异常情况，不涉及程序修改、示教或维修操作；
   - 操作人员不得在安全围栏内进行作业。
2. 编程人员：
   - 进行机器人程序的编写、修改、优化及存储管理；
   - 在指定安全区域内，通过示教器完成机器人的示教编程、轨迹调试等作业；
   - 验证程序运行效果，确保程序符合作业需求，不涉及机器人核心部件维修或结构调整；
   - 编程人员必须接受针对机器人的专业培训。
3. 维修人员：
   - 执行机器人的定期维护、故障排查、部件修理、参数调整及零件更换作业；
   - 为维修作业需要，可在安全区域内进行必要的机器人辅助示教或操作；
   - 记录维护维修数据，反馈设备运行损耗情况，确保机器人性能稳定；
   - 维修人员必须接受针对机器人的专业培训。

使用人员权限划分：本表格界定安全围栏外作业人员的操作权限范围；符号 “O”代表该人员具备对应操作权限，“-” 代表无操作权限。

表 0-1 安全围栏外作业权限一览表

作业内容	操作人员	编程人员	维修人员
机器人控制柜电源通断	O	O	O
选择自动/远程（A)、手动（T1）、手动100%（T2）操作模式	-	O	O

表 0-1 安全围栏外作业权限一览表 (续)

作业内容	操作人员	编程人员	维修人员
示教器选择机器人程序	-	O	O
外部设备选择机器人程序	-	O	O
示教器启动机器人程序	O	O	O
示教器复位报警	-	O	O
示教器参数设置	-	O	-
示教器示教操作	-	O	O
控制柜急停按钮操作	O	O	O
示教器急停按钮操作	O	O	O
控制面板维护	-	O	-
维护期间示教操作	-	-	O

所有涉及机器人操作、编程调试及维护维修的作业场景，操作人员、编程人员、维修人员必须严格遵守安全防护要求，全程规范穿戴符合安全标准的防护装备：

- 需选用与作业内容适配的作业服，避免衣物卷入设备或勾挂部件；
- 必须穿戴具备防砸、防滑功能的安全鞋；
- 作业期间需正确佩戴安全帽；
- 需定期检查防护装备完好性，破损、老化的装备需立即更换，若涉及电气维修、高温作业等特殊场景，还需按对应作业规范额外配备绝缘手套、耐热手套等专项防护装备。

说明

关于培训的更多信息，请咨询埃斯顿公司工作人员。

安全标示

本手册中出现的以下安全标示，均为核心操作与安全指引依据，用户在阅读手册内容时，必须仔细研读标示对应的说明，严格遵守相关要求，规避安全风险。

标示符号	定义说明
 危险	• 本标志对应的文本表示存在高度潜在危险。若无法避免相关风险，将直接导致人员死亡或严重伤害，必须严格遵守相关安全要求。
 警告	• 本标志对应的文本表示存在中度或低度潜在危险。若忽视相关提示，可能导致人员轻微伤害或中等程度伤害，需规范操作规避风险。
 注意	• 本标志对应的文本表示存在非人身伤害类潜在风险。若忽视相关说明，可能造成设备损坏、数据丢失、设备性能下降或其他不可预知的不良后果。

标示符号	定义说明
 说明	• 本标志对应的文本为正文的附加补充信息，用于强调核心内容、补充操作细节或解释关键概念，帮助用户更准确理解手册内容并高效完成操作。

使用人员的安全事项

1. 搬运与安装机器人时，必须严格遵循埃斯顿公司规定的标准流程操作。采用非标准方法可能导致机器人倾倒、部件损坏，进而引发安全事故。
2. 机器人安装前必须划定安全防护区域，需在其工作区域周边设置防护栅栏及安全警示牌。
3. 机器人上方严禁放置悬挂物，防止物体掉落砸坏机器人及周边设备。
4. 严禁倚靠控制柜或随意触碰按钮，避免机器人产生意外动作，造成人身伤害或设备损坏
5. 拆分机器人过程中，需警惕设备上松动或脱落的零件，防止砸伤作业人员。
6. 对外部设备进行单独调试时，必须先断开机器人电源，再开展操作。
7. 所有外围设备必须连接符合规范的接地线，确保用电安全。
8. 首次操作机器人时，必须先以低速运行，再逐步提升速度，全程观察并确认设备无异常。
9. 使用示教器时，戴手套可能引发操作失误，必须摘下手套再进行操作。
10. 程序、系统变量等关键信息可存储至存储卡等介质，为防止意外导致数据丢失，建议用户定期备份数据。
11. 严禁擅自搬动机器人各轴，否则可能引发人身伤害或设备损坏。
12. 进行控制柜与机器人、外围设备间的配线及配管时，必须采取防护措施（如将管线、线缆从预埋坑穿过或加装保护盖遮挡），避免被人员踩踏或叉车碾压损坏。
13. 机器人工作期间均可能产生意外动作，对工作范围内人员造成严重伤害或损坏设备。机器人启动前，必须测试安全门、抱闸、安全指示灯等所有安全措施的可靠性，且必须确保工作范围内无其他人员。
14. 通过软件设定的动作范围及负载条件，严禁设定超出产品规格表中的规定值；设置不当可能引发人员伤害或设备损坏。
15. 若必须在机器人工作范围内作业，需遵循以下安全规则：
    - 需将机器人切换至手动模式后再通过示教器上使能，并断开上位机控制等所有自动控制方式；
    - 机器人处于手动模式时，运行速度需限制在 250mm/s 以下；若需调至手动全速度，仅允许充分了解风险的专业人员操作；
    - 留意机器人转动关节，防止头发、衣物卷入；同时需警惕机器人及其他附属设备运动可能引发的其他危险（如碰撞、挤压等）；
    - 作业前需测试电机抱闸功能是否正常，防止因机器人异常运动导致人身伤害；

- 需预先规划机器人突然向自身方位运动时的应急避让方案；
- 需预设安全躲避区域，以备紧急情况使用。

危险

在任何工况下，操作人员、编程人员及维修人员均严禁在机器人本体机械臂下方站立、停留或进行作业。该区域存在极高安全风险：机器人可能因系统故障发生异常运动，或被他人误触使能开关启动，导致人员被碰撞、挤压，造成重伤甚至死亡。所有相关人员必须严格遵守此禁令，确保作业安全。

注意

机器人作业现场及控制柜周边，必须配备二氧化碳灭火器（≥2kg），确保灭火器放置在易取用、无遮挡位置。

操作人员的安全注意事项：

1. 操作机器人前，必须先按下控制柜前门及示教器右上方的急停按钮，检查 “伺服准备”指示灯是否熄灭，确认机器人电源已完全关闭。
2. 操作期间，严禁非工作人员触碰或操作控制柜。否则可能导致机器人产生意外动作，引发人身伤害或设备损坏。
3. 向机器人安装工具时，必须先关闭（OFF）控制柜及待装工具的电源，并悬挂警示牌。安装期间若擅自接通电源，可能造成触电，或导致机器人意外运动引发伤害。
4. 急停功能：
   - 急停功能独立于机器人所有电气控制，可强制停止机器人全部运动。
   - 触发急停后，驱动机器人上伺服电机运行的动力电源将会断开，伺服电机抱闸，机器人停止运动；
     需先释放急停按钮，再重新启动机器人，设备方可恢复运行。

急停按钮示意图	特别说明
	机器人系统配备多个急停按钮，用于紧急停止机器人运行：其中示教器与电控柜各标配 1 个红色急停按钮，用户也可根据实际需求额外增设。 所有急停按钮均需安装在易于触及的位置，确保突发意外时能快速触发，及时停止机器人运行。

危险

操作人员需要注意伺服电机的动力线、连接夹具和其他装置的动力线的高电压危险。

⚠ 注意

急停功能仅用于紧急场景（如人身安全受威胁、设备突发失控等）下强制停止机器人，严禁用于日常程序暂停、正常关机等非紧急操作。

编程人员的安全注意事项：

机器人示教作业期间，编程人员若需进入机器人动作范围，必须重点关注作业安全。

⚠ 注意

自动模式下，使能的接通和断开通过操作示教器上的Mot按钮实现切换：若当前为使能断开状态，按下按钮后将会切换到使能接通状态；反之，当前为使能接通状态，按下按钮后则会切换到使能断开状态。对于正在运行的机器人，需要先按下Stop按键停止运行，之后再按下Mot按钮实现使能断开。

安全使用示教器，必须遵守以下规则：

- 确保使能按钮始终处于有效状态，避免应急时无法正常操作。
- 暂停机器人运行、编程或测试期间，需及时断开使能，防止误触发。
- 示教者进入机器人工作区域时，必须随身携带示教器，避免他人在不知情时操作机器人。
- 禁止将示教器放置于机器人工作范围内，防止机器人运动时碰撞示教器，引发异常动作。

维修人员的安全注意事项：

1. 机器人发热部件安全警示：机器人正常运行时，部分关键部件会发热，其中伺服电机、减速机的发热尤为明显，靠近或直接触碰易导致烫伤。若需在部件发热状态下触碰，必须佩戴耐热手套等防护用具，避免直接接触高温部件。

   ⚠ 注意

   触摸这些高温部件前，需先用手靠近部件表面初步感受温度，判断是否安全，防止直接触摸导致烫伤。

   机器人停机后，需等待高温部件冷却至安全温度，再开展维修工作，避免高温引发人员灼伤。

2. 拆卸部件的安全注意事项：拆卸前必须先确认齿轮、轴承等内部零件完全停止旋转与运动，再打开部件盖子或保护装置；严禁在上述部件旋转时开启保护装置，防止卷入风险。若内部存在活动零件（非固定状态），需使用辅助装置将其固定在原始位置，避免拆卸过程中零件移位、掉落引发安全隐患。
   维修、安装或保养后的首次启动测试，必须严格遵循以下步骤：

a. 彻底清理机器人本体及其工作区域内的所有工具、材料及临时装置，确保无遗留物；

b. 完整恢复并确认所有安全防护装置已正确安装且功能正常；

c. 确认所有人员已撤离至机器人安全工作范围之外，禁止任何人员在测试期间进入作业区域；

d. 启动测试时，应重点监控本次维修或更换部件的运行状态，实时观察是否存在异响、卡顿等异常，及时停机排查。

⚠ 注意

在维修过程中，禁止站立、攀爬或倚靠机器人任何部位。必须使用符合安全标准的登高设备，以防跌落造成严重人身伤害。

---

3. 气动/液压系统安全注意事项：关闭气源或液压泵后，系统内仍会残留带压气体或液体，此类残留物具有潜在能量。需通过专项措施释放残留能量，避免其对人员或设备造成冲击、喷射等伤害；维修气动或液压元件前，必须确保系统内残留能量已完全释放。

⚠ 注意

为防止气动/液压系统能量意外释放引发安全事故，必须安装安全阀，作为系统压力过载时的强制保护装置。

---

4. 故障诊断阶段需开启机器人电源，但维修机器人时必须完全关闭电源，并切断所有相关电源连接，避免带电作业引发安全风险。
5. 抱闸检测：机器人正常运行时，抱闸易因磨损导致锁止性能下降，需定期开展功能检测以保障关节安全锁止。具体检测步骤如下：

   a. 将机器人各关节调整至承受最大负载的姿态，模拟极端受力场景。

   b. 关闭机器人主电源，使抱闸自动生效。

   c. 在各关节的固定部件与活动部件间做对位标记，确保标记清晰可辨，便于后续观察位移。

   d. 静置规定时长（通常为 10-30 分钟）后，观察各关节的对位标记是否偏移，以此判断抱闸是否存在滑闸、锁止失效情况。
6. 向减速机加注润滑油时，存在高温油液烫伤、压力溅射、油液污染等风险，可能伤及人员或损坏设备。开展加油作业前，必须严格遵守以下安全规范：

   - 加注或排放润滑油时，需佩戴防化手套、护目镜等防护装备。
   - 打开油腔盖前需确认油腔温度降至安全范围（建议≤40℃），操作时保持安全距离（建议≥50cm）；油腔内可能残留压力，需缓慢开启盖子，避免油液溅射引发伤害。

- 按油量表刻度加注润滑油，严禁超过最大刻度线；加注完成后，需检查油液指示口，确认油位处于标准范围内。
- 严禁将不同型号、不同牌号的润滑油混入同一减速机；更换润滑油型号前，需用专用清洁剂彻底冲洗油腔，确保残余旧油清理干净。
- 排放旧油时，需将油腔内旧油彻底排空（可借助负压设备辅助）；加油后需再次检查油液指示口，双重确认油位符合要求。

⚠ 注意

放空减速机旧油前，可先短时间低速运行机器人（建议 5-10 分钟，使油液温度升至 40-60℃），通过加热降低油液黏度，让放油更顺畅，同时减少油腔内壁的旧油残留。需注意控制运行时长，避免油液温度过高（不超过 60℃），防止后续操作时因高温油液引发烫伤。

刀具及外围设备安全注意事项

机器人关闭后，其连接的刀具、外围设备可能仍独立运行（未随机器人断电）。若此类设备的电源线、动力线缆出现破损，或人员误触带电部件，易引发触电等人身伤害。因此，操作前必须单独切断刀具及外围设备的电源，并检查线缆完整性，再开展后续作业。

机器人手臂紧急救援安全规范

当发生机器人手臂夹伤操作人员的紧急情况时，需立即对夹持手臂进行紧急移除，具体操作及安全要求如下：

1. 紧急移除操作：
   - 小型机器人手臂：可采用手动移除方式，需 2 人及以上协作（1 人固定手臂、1 人协助受困者脱困），严禁单人操作以防手臂晃动加重伤害。
   - 大型机器人手臂：必须使用吊车、液压千斤顶等专业承重设备，严禁使用小型推车、撬棍等非承重工具，避免设备倾覆或手臂坠落引发二次事故。
2. 释放关节抱闸前，必须先通过专用支架、承重绳索等工具稳固固定机械臂，确保机械臂完全脱离 “悬空承重” 状态，防止释放抱闸后手臂因重力下坠，对受困者造成挤压、砸伤等二次伤害。

⚠ 警告

移除过程中若遇机械锁死、部件卡顿等问题，需立即停止操作并联系埃斯顿公司技术人员获取专业指导，严禁擅自拆解关节、抱闸等关键部件，避免破坏设备结构或引发新风险。

机器人停止方式及操作规范

停止方式	核心定义	执行流程	安全注意事项
断电停止（紧急断电式）	通过断开伺服电源实现机器人瞬间停止的方式，停止过程中减速轨迹不受控制，仅适用于紧急场景。	1. 触发断电停止（如急停按钮）后，系统立即发出报警信号并断开伺服电源，机器人动作瞬间终止； 2. 同步暂停当前程序的执行。	1. 严禁在机器人正常动作时频繁使用断电停止，否则易导致伺服系统、机械结构故障； 2. 日常操作中应避免将断电停止作为常规停止方式，仅在紧急避险时使用。
报警停止（故障触发式）	机器人系统因过载、故障等（断电报警除外）触发报警后，通过控制指令实现减速停止的方式，停止过程可控。	1. 机器人系统检测到过载、部件故障等异常后，触发非断电类报警； 2. 伺服系统自动发出 “控制停止” 指令，机器人按预设减速轨迹平稳停止； 3. 程序执行暂停，随后系统断开伺服电源。	报警停止触发后，需先排查并排除故障根源，确认设备状态正常后再重启运行，避免故障扩大。
保持停止（伺服维持式）	维持伺服电源接通，通过系统控制使机器人减速停止的方式，停止后设备仍处于待机状态。	1. 触发保持停止指令后，机器人按减速逻辑平稳停止动作； 2. 程序执行暂停，伺服电源保持接通，设备维持待机状态。	停止后若需进行设备调试、位置调整，可直接通过示教器操作，无需重新启动伺服系统。 ⚠ **注意** 操作前需确认作业区域无干涉。

目录

申明.....2
前言.....3
安全使用须知.....4
表目录.....19
1 库功能介绍.....20
1.1 项目背景.....20
1.2 设计说明.....20
1.3 功能列表.....21
1.4 模式权限说明.....24
2 API功能库介绍.....26
2.1 API库目录结构.....26
3 使用操作.....27
3.1 C++工程搭建.....27
3.1.1 环境配置要求.....27
3.1.2 操作步骤.....27
3.1.3 C#工程搭建.....30
3.1.3.1 环境配置要求.....30
3.1.3.2 操作步骤.....31
3.2 VB工程搭建.....34
3.2.1 环境配置要求.....34
3.2.2 操作步骤.....35
4 运行环境.....39
4.1 软件支持.....39
4.2 示教器端配置.....39
5 典型应用案例.....41
5.1 上位机加载机器人示教程序并再现运行.....41
5.2 上位机规划点位控制机器人运动.....42
6 接口参考.....44
6.1 结构化数据.....44
6.1.1 内部字符串.....44
6.1.2 权限信息.....45
6.1.3 坐标类型.....45
6.1.4 系统模式.....46
6.1.5 运动模式.....46
6.1.6 IO类型.....47
6.1.7 变量作用域类型.....47

6.1.8 用户坐标系类型.....48
6.1.9 报警等级.....48
6.1.10 运动状态.....48
6.1.11 使能状态.....49
6.1.12 安装方式.....49
6.1.13 震动抑制等级.....49
6.1.14 指令标识.....50
6.1.15 工程程序名.....50
6.1.16 附加指令.....50
6.1.17 附加指令结构一.....51
6.1.18 空间坐标形态配置数据.....51
6.1.19 机器人位置.....53
6.1.20 机器人速度.....56
6.1.21 机器人过渡区域.....57
6.1.22 负载参数.....59
6.1.23 用户坐标系.....61
6.2 接口列表.....64
6.2.1 默认构造函数.....64
6.2.2 有参构造函数.....64
6.2.3 有参构造函数.....64
6.2.4 设置本地日志配置.....65
6.2.5 日志打印开关.....66
6.2.6 写本地日志.....66
6.2.7 错误日志.....67
6.2.8 连接服务器.....67
6.2.9 断开连接.....68
6.2.10 连接检测.....68
6.2.11 强制开启API模式接口.....69
6.2.12 获取控制器时间接口.....69
6.2.13 获取当前模式的工作状态接口.....70
6.2.14 申请机器人运动操作权限接口.....70
6.2.15 查询当前机器人运动操作权限接口.....71
6.2.16 释放机器人运动操作权限接口.....71
6.2.17 唤醒机器人运动接口.....72
6.2.18 暂停机器人运动接口.....72
6.2.19 恢复机器人运动接口.....73
6.2.20 停止机器人运动接口.....73
6.2.21 获取机器人运动状态接口.....74
6.2.22 运动下发（协议接口）接口.....74

6.2.23 执行MovJ指令接口 75
6.2.24 执行MovJ2指令接口 75
6.2.25 执行MovL指令接口 76
6.2.26 执行MovL2指令接口 77
6.2.27 执行MovC指令接口 78
6.2.28 执行MovC2指令接口 79
6.2.29 执行MovArch指令接口 80
6.2.30 执行MovArch3指令接口 81
6.2.31 执行MovArch2指令接口 82
6.2.32 执行E_MovJArch2指令接口 82
6.2.33 执行E_MovLArch2指令接口 83
6.2.34 获取所有工具坐标系ID接口 84
6.2.35 创建工具坐标系接口 85
6.2.36 修改工具坐标系接口 85
6.2.37 加载工具坐标系接口 86
6.2.38 获取当前工具ID接口 86
6.2.39 获取工具坐标变量名称接口 87
6.2.40 获取工具坐标系变量接口 87
6.2.41 获取所有用户坐标系ID接口 88
6.2.42 创建用户坐标接口 88
6.2.43 修改用户坐标接口 89
6.2.44 加载用户坐标接口 89
6.2.45 获取当前用户坐标ID接口 90
6.2.46 获取用户坐标变量名称接口 90
6.2.47 获取用户坐标信息接口 91
6.2.48 获取所有负载ID接口 91
6.2.49 创建负载接口 92
6.2.50 修改负载接口 92
6.2.51 加载负载接口 93
6.2.52 获取当前负载变量ID接口 93
6.2.53 获取负载变量名称接口 94
6.2.54 获取负载变量接口 94
6.2.55 获取默认加载的负载参数接口 95
6.2.56 设置IO的值接口 95
6.2.57 设置IO的值（带等待）接口 96
6.2.58 获取IO的值接口 96
6.2.59 设置IO备注接口 97
6.2.60 获取IO备注接口 98
6.2.61 刷新全局变量接口 98

6.2.62 设置变量是否掉电保存接口 99
6.2.63 设置变量接口 99
6.2.64 获取变量接口 100
6.2.65 设置默认位置到精度值接口 101
6.2.66 获取默认位置到精度值接口 101
6.2.67 设置点动坐标系类型接口 101
6.2.68 获取点动坐标系类型接口 102
6.2.69 设置jog是否可用接口 102
6.2.70 获取jog是否可用接口 103
6.2.71 点动接口 103
6.2.72 寸动接口 104
6.2.73 示教点接口 104
6.2.74 获取机器人类型接口 105
6.2.75 获取机器人信息接口 105
6.2.76 获取本体轴数接口 106
6.2.77 获取附加轴数接口 106
6.2.78 获取机器人状态接口 106
6.2.79 设置全局速度接口 107
6.2.80 获取全局速度接口 107
6.2.81 设置系统模式接口 108
6.2.82 获取系统模式接口 108
6.2.83 获取急停按钮状态接口 109
6.2.84 设置运动模式接口 109
6.2.85 获取运动模式接口 110
6.2.86 设置机器人伺服使能接口 110
6.2.87 获取机器人伺服是否使能接口 110
6.2.88 获取机器人待机励磁状态接口 111
6.2.89 设置震动抑制使能接口 111
6.2.90 获取震动抑制使能状态接口 112
6.2.91 设置震动抑制等级接口 112
6.2.92 获取震动抑制等级接口 113
6.2.93 清除伺服报警接口 113
6.2.94 设置伺服参数接口 113
6.2.95 获取伺服参数接口 114
6.2.96 操作伺服参数接口 114
6.2.97 伺服备份（C1 控制柜单轴）接口 115
6.2.98 伺服恢复（C1 控制柜单轴）接口 115
6.2.99 获取伺服信息接口 116
6.2.100 获取伺服历史报警接口 116

6.2.101 获取伺服状态接口.....117
6.2.102 设置碰撞检测使能接口.....118
6.2.103 获取碰撞检测使能状态接口.....118
6.2.104 设置碰撞检测参数接口.....118
6.2.105 获取碰撞检测参数接口.....119
6.2.106 单轴回零接口.....119
6.2.107 整体回零接口.....120
6.2.108 获取单圈值接口.....120
6.2.109 设置机器人安装方式接口.....121
6.2.110 获取机器人的安装方式接口.....121
6.2.111 获取是否支持吊装接口.....121
6.2.112 向机器人推送报警信息接口.....122
6.2.113 获取当前错误个数接口.....122
6.2.114 获取指定的错误信息接口.....123
6.2.115 清除所有错误报警接口.....123
6.2.116 清除所有错误报警后获取是否还有错误接口.....124
6.2.117 清除错误报警（指定错误号）接口.....124
6.2.118 获取机器人当前错误号（最近触发）接口.....125
6.2.119 获取机器人错误内容（默认语言）接口.....125
6.2.120 获取机器人错误内容（指定语言）接口.....125
6.2.121 获取当前位置信息（关节坐标系）接口.....126
6.2.122 获取当前位置信息（用户坐标系）接口.....126
6.2.123 获取当前位置信息（世界坐标系）接口.....127
6.2.124 获取当前位置信息（用户坐标系，带时间戳）接口.....127
6.2.125 获取当前位置信息（世界坐标系，带时间戳）接口.....128
6.2.126 设置当前外部坐标系（按变量名）接口.....128
6.2.127 设置当前外部坐标系（按 ID）接口.....129
6.2.128 获取当前加载的程序名接口.....129
6.2.129 获取当前程序运行状态接口.....129
6.2.130 卸载工程接口.....130
6.2.131 卸载程序接口.....130
6.2.132 只加载工程接口.....131
6.2.133 加载程序接口.....131
6.2.134 刷新变量依赖接口.....132
6.2.135 设置自启动程序接口.....132
6.2.136 加载自动加载程序接口.....133
6.2.137 设置程序运行指针位置接口.....133
6.2.138 获取程序指针接口.....133
6.2.139 运行程序接口.....134

6.2.140 暂停程序接口 134
6.2.141 停止程序接口 135
6.2.142 读取 Int 型变量值接口 135
6.2.143 写入 Int 型变量值接口 136
6.2.144 读取 Double 型变量值接口 136
6.2.145 写入 Double 型变量值接口 137
6.2.146 读取 String 型变量值接口 137
6.2.147 写入 String 型变量值接口 138
6.2.148 读取位置类型变量值接口 138
6.2.149 写入位置类型变量值接口（E_WritePos/E_WritePos2） 139
6.2.150 设置 DO 的值接口 139
6.2.151 获取最近一次下发成功的运动指令的信息接口 140
6.2.152 获取最近一次下发成功的运动指令的执行状态接口 140
6.2.153 获取指定运动指令执行状态接口 141
6.2.154 接口耗时接口 141
6.2.155 设置点位数组接口 141
6.2.156 获取点位数组接口 142
6.2.157 获取力矩的值接口 143
6.2.158 设置控制器网络配置接口 143
6.2.159 获取控制器网络配置接口 144
6.2.160 获取 IO 个数接口 144
6.2.161 计算用户坐标系接口 145
6.2.162 计算工具坐标系_4 点法接口 146
6.2.163 计算工具坐标系_1 点法接口 146
6.2.164 执行 MovJ2 数组指令接口 147
6.2.165 执行 MovJOffset 指令接口 147
6.2.166 执行MovJ2 数组指令（独立配置速度 / 过渡参数）接口 148
6.2.167 执行 MovL2 指令数组接口 149
6.2.168 执行MovLOffset 指令接口 150
6.2.169 执行MovL2 指令数组（独立配置参数）接口 150
6.2.170 执行 MovJ3 运动指令接口 151
6.2.171 执行 MovL3 运动指令接口 152
6.2.172 获取指定运动指令执行状态（MovJ3/MovL3 专用）接口 152
6.2.173 获取运动队列中尚未完成的运动段数量接口 153
6.2.174 获取指定运动数组指令执行状态接口 153
6.2.175 获取指定运动数组指令的子运动 ID接口 154
6.2.176 笛卡尔坐标转关节坐标接口 154
6.2.177 关节坐标转笛卡尔坐标接口 155
6.2.178 获取机器人当前线速度接口 155

6.2.179 弧焊通用协议设置指令接口.....156
6.2.180 弧焊通用协议获取指令接口.....156
6.2.181 弧焊状态获取指令接口.....157
6.2.182 设置世界坐标系加减速度接口.....157
6.2.183 设置关节坐标系加减速度接口.....158
6.2.184 设置轴运动参数接口.....158
6.2.185 获取轴运动参数接口.....159
6.2.186 设置轴限位参数接口.....159
6.2.187 获取轴限位参数接口.....160
6.2.188 获取机器人 Jog 状态接口.....160
6.2.189 设置 PSO 控制字使能与禁能指令接口.....161
6.2.190 获取 PSO 状态字的值接口.....161
6.2.191 设置指定轴的指定 PSO 比较点指令接口.....162
6.2.192 执行 MovCircle 指令接口.....162
6.2.193 执行 MovCircle2 指令接口.....163
6.2.194 判断当前运动指令是否全部完成接口.....164
6.2.195 机器人停止清空运动队列接口.....164
6.2.196 停止机器人并清空队列接口.....165
6.2.197 SCARA 机型 3 轴抱闸控制接口.....165
6.2.198 获取抱闸状态接口.....166
6.2.199 笛卡尔坐标转换（坐标系映射）接口.....166
A.1 错误列表.....168

表目录

表 1-1 主控权限.....24
表 1-2 监控权限.....25

1 库功能介绍

介绍功能库的基本框架和主要功能。

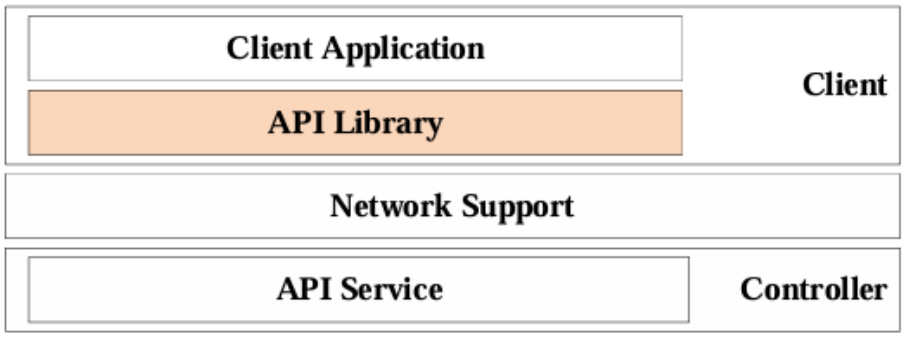
1.1 项目背景

为充分满足客户对机器人运动控制的定制化需求，以及更好的与客户自身控制逻辑或产品相结合，埃斯顿机器人提供了一套完整的接口调用库（以下简称API库）供客户进行二次开发来实现对机器人的控制。

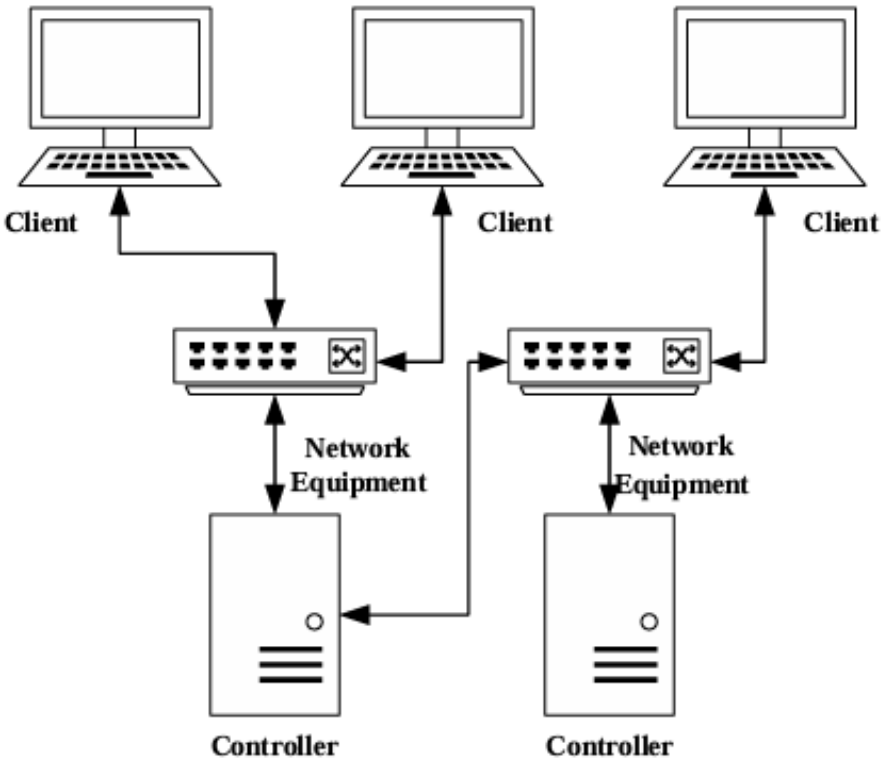
API库提供了多种开发语言支持，包括C++，C#和VB，适应不同客户的开发需求。

1.2 设计说明

API库工作于客户上位机，本身不具备单独运行能力，仅提供用户二次开发以及运行依赖支持。用户基于API库完成应用程序开发，运行程序时会依赖API库对象，通过API库提供的接口，完成与控制器上服务端的数据交互，达到驱动机器人的目的。



在功能实现上，API库提供了接口操作类，通过创建类实例与控制器服务连接，客户程序可以同时创建多个类实例与多台控制器连接，控制器服务也可以同时接受多个客户端连接。提供了客户程序与机器人的多对多连接关系，既支持多个客户机程序同时与一台机器人控制器连接，也支持一台客户机程序同时连接多台机器人控制器。



1.3 功能列表

API库提供了如下的功能接口，用于满足客户对机器人的运动控制以及状态监视的需求。

序号	功能分类	功能名称	详细描述
1	基本功能	日志打印开关	是否输出本地日志
2		写本地日志	将日志信息写入本地文件
3		错误日志	获取最近一次错误的信息
4		连接服务器	连接服务器
5		断开连接	断开与服务的连接
6		连接检测	检测是否已连接服务
7	系统控制	申请权限	申请机器人运动操作权限
8		查询权限	查询当前机器人运动操作权限
9		释放权限	释放机器人运动操作权限
10		使能开关	给机器人上/下励磁
11		获取使能状态	获取机器人是否上励磁
12		jog使能开关	设置当前模式下job是否可用
13		获取jog使能状态	获取当前模式下job是否可用
14		设置到位精度	设置默认到位精度值，掉电保存
15		获取到位精度	获取默认到位精度值
16		设置jog坐标类型	设置点动坐标系类型，轴关节坐标或世界坐标

序号	功能分类	功能名称	详细描述
17		获取jog坐标类型	获取点动坐标系类型，轴关节坐标或世界坐标
18		获取机器人类型	获取机器人型号信息
19		获取机器人信息	获取机器人信息，一些附加备注信息
20		获取机器人在线状态	获取机器人状态，目前仅有连接状态
21		获取控制器状态	获取控制器当前状态，有正常，错误和停止三种状态
22		设置全局速度	设置机器人运行的速度百分比
23		获取全局速度	获取机器人运行的速度百分比
24		获取系统模式	获取机器人当前的工作模式
25		设置运动模式	设置机器人当前的运动模式，单步或连续
26		获取运动模式	获取机器人当前的运动模式
27		清除错误	清除下位机报告的机器人错误警告信息
28		获取错误号	获取错误警告编号
29		获取错误详情	根据错误号，获取错误详细内容，支持中英文
30		获取关节坐标	获取机器人当前位置信息，关节坐标系
31		获取世界坐标	获取机器人当前位置信息，世界坐标系
32	运动控制	开始	恢复机器人运动，从停止模式转到运行模式
33		暂停	暂停机器人运动，不会清空运动队列
34		继续	恢复机器人运动，继续执行运动队列
35		停止	停止机器人运动，同时清空运动队列
36		点动	使机器人按照轴关节或空间坐标进行运动，一次运动微量距离
37		寸动	使机器人按照轴关节或空间坐标方式运动，一次运动指定距离
38		MovJ	关节运动指令，参数为机器人内部变量名称
39		MovJ2	关节运动指令，参数为对应类型的对象
40		MovL	直线运动指令，参数为机器人内部变量名称
41		MovL2	直线运动指令，参数为对应类型的对象
42		MovC	圆弧运动指令，参数为机器人内部变量名称
43		MovC2	圆弧运动指令，参数为对应类型的对象
44		MovArch	拱形运动指令，参数为机器人内部变量名称
45		MovArch2	拱形运动指令，参数为对应类型的对象
46		获取运行指令号	获取最近一次下发成功的运动指令的信息
47		获取指令状态	获取最近一次下发成功的运动指令的执行状态
48		获取指令状态	获取指定运动指令执行状态
49		获取运动状态	获取机器人当前的运动状态：运动/停止
50	参数设置	设置外部坐标系	根据外部坐标系变量名设置系统当前使用的外部坐标系
51		设置外部坐标系	根据外部坐标系ID设置系统当前使用的外部坐标系

序号	功能分类	功能名称	详细描述
52		新建工具参数	在系统内创建指定标号的工具参数
53		修改工具参数	修改指定标号的工具参数
54		选择工具参数	设置系统当前使用的工具参数
55		获取当前工具号	获取当前加载的工具号
56		获取工具参数	根据工具号获取工具的参数值
57		新建用户坐标	在系统内创建指定标号的用户坐标
58		修改用户坐标	修改指定标号的用户坐标
59		选择用户坐标	设置系统当前使用的用户坐标
60		获取当前坐标号	获取当前加载的用户坐标号
61		获取坐标参数	根据坐标号获取坐标的参数值
62		新建负载参数	在系统内创建指定标号的负载参数
63		修改负载参数	修改指定标号的负载参数
64		选择负载参数	设置系统当前使用的负载参数
65		获取当前负载号	获取当前加载的负载号
66		获取负载参数	根据负载号获取负载的参数值
67	IO设置	设置IO的值	设置系统IO的值，包括DI,DO,AI,AO,SimDI,SimDO,SimAI,SimAO
68		获取IO的值	获取IO的值，包括DI,DO,AI,AO,SimDI,SimDO,SimAI,SimAO
69		设置DO的值	修改系统DO的值
70	变量设置	设置变量	设置或增加变量，支持数字，字符，位置等常规类型
71		获取变量	获取指定变量的值
72		获取整形变量	获取指定变量的值
73		设置整形变量	设置或增加变量
74		获取实数变量	获取指定变量的值
75		设置实数变量	设置或增加变量
76		获取字符变量	获取指定变量的值
77		设置字符变量	设置或增加变量
78		获取位置变量	获取指定变量的值
79		设置位置变量	设置或增加变量
80		示教点	将当前机器人坐标值赋值给位置变量
81	程序运行	获取当前程序	获取系统当前加载的程序信息，包括工程名、程序名
82		获取程序状态	获取当前程序运行状态，运行、暂停或停止
83		卸载工程	卸载工程，连带程序一起卸载，系统恢复到未加载程序的状态
84		卸载程序	卸载程序，工程依旧加载
85		加载工程	只加载工程，不会加载工程内程序
86		加载程序	加载工程和程序

序号	功能分类	功能名称	详细描述
87		运行程序	机器人执行加载的程序
88		暂停程序	机器人暂停执行加载的程序
89		停止程序	机器人停止执行加载的程序

1.4 模式权限说明

API库在所有模式下都可以调用，为满足安全需要，在不同模式下，API的功能会由不同限制，具体参考下表：

说明

在停止状态，非API模式下，才可以切换到API模式。

表 1-1 主控权限

功能名		功能描述	API接口支持	非API模式		
				远程	自动	手动
			API模式			
工程操作接口		操作用户工程的接口，加载、注销、工程启动、工程暂停、工程停止等	X	X	√	√
变量操作	全局变量	全局变量的操作接口，读、写、创建、删除等操作	√	√	√	√
	工程变量、程序变量	工程、程序变量的操作接口，读、写、创建、删除等操作	X	X	√	√
运动指令		运动指令MovJ\MovL等接口函数，以及运动中设置精度，不能限制速度，运动暂停等功能	√	X	X	X
Jog指令		Jog机器人的接口（设置Jog坐标系，点动、寸动等jog机器人用到的接口）	√	X	X	√
机器人状态		机器人速度、使能、坐标系、当前系统状态、当前错误等等机器人的各种状态（好的话能都开放）	√	√	√	√
IO指令	实际IO	实际IO读写操作接口	√	√	√	√
	虚拟IO	虚拟IO读写操作接口	√	√	√	√

表 1-2 监控权限

功能名		功能描述	API模式	非API模式		
				远程	自动	手动
			API接口支持			
工程操作接口		操作用户工程的接口，加载、注销、工程启动、工程暂停、工程停止等	X	X	√	√
变量操作	全局变量	全局变量的操作接口，读、写、创建、删除等操作	√	√	√	√
	工程变量、程序变量	工程、程序变量的操作接口，读、写、创建、删除等操作	X	X	√	√
运动指令		运动指令MovJ\MovL等接口函数，以及运动中设置精度，不能限制速度，运动暂停等功能	X	X	X	X
Jog指令		Jog机器人的接口（设置Jog坐标系，点动、寸动等jog机器人用到的接口）	√	X	X	√
机器人状态		机器人速度、使能、坐标系、当前系统状态、当前错误等等机器人的各种状态（好的话能都开放）	√	√	√	√
IO指令	实际IO	实际IO读写操作接口	√	√	√	√
	虚拟IO	虚拟IO读写操作接口	√	√	√	√

2 API功能库介绍

介绍API功能库的文件目录结构。

2.1 API库目录结构

解压库文件。解压完成后，在指定目录会生成三个目录。

文件名	用途	备注
help	代码示例工程	一个vs解决方案，包含c++，c#和vb工程
x32	32位API库	包含头文、lib库、运行依赖库以及一个测试程序
x64	64位API库	同上

库文件目录详细说明，x32和x64结构一致，此处以x32举例。

文件名	用途
bin/EstunApiFroPublic-vc140-x32.dll	.Net项目引用库
bin/EstunRemoteApiLib-vc140-x32.dll	C++/.Net程序运行依赖库
Include/EstunRemoteApiLib	C++项目需要包含的头文件
lib/EstunRemoteApiLib-vc140-x32.lib	C++项目的附加依赖库

3 使用操作

介绍基于API库自定义工程，实现对机器人的控制。

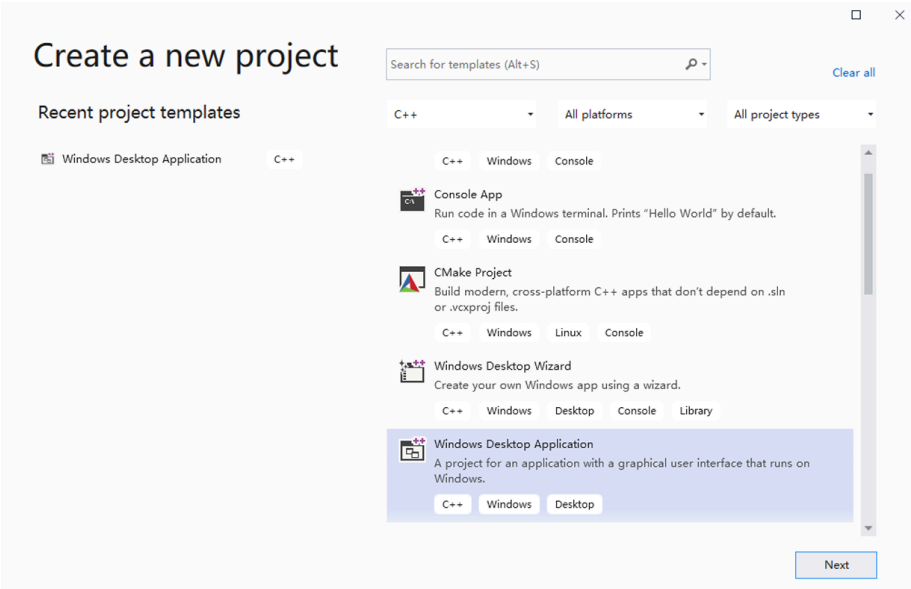
3.1 C++工程搭建

3.1.1 环境配置要求

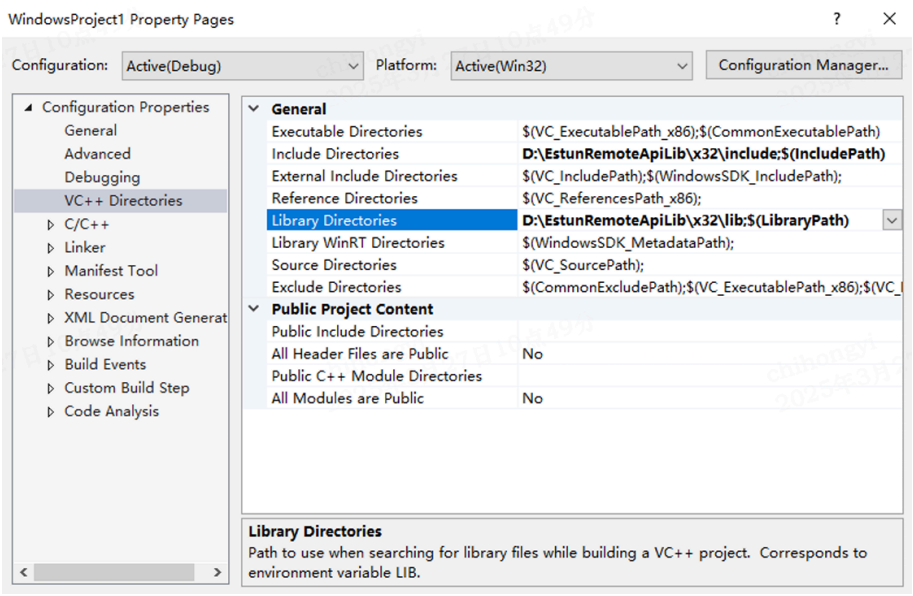
配置项	最低要求	推荐配置	备注
系统版本	Win7	Win10	-
内存大小	4GB	8GB	-
VisualStudio版本	2015	2017	-
QT版本	4.8.5	5.11	开发基于QtGui的程序时需要配置该配置项

3.1.2 操作步骤

1. 新建工程。
   打开vs2015，单击左上角“**文件 -> 新建 -> 项目**”，选择“**Visual C++ -> Windows桌面应用工程**”。



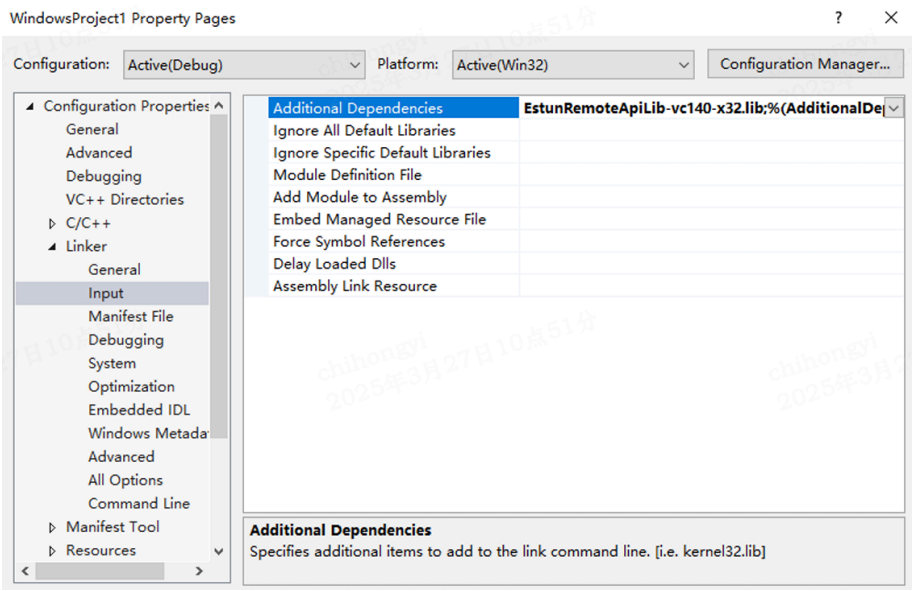
2. 配置头文件路径和库路径。
   右键单击工程，单击**“属性”**，选择“**配置属性 -> VC++目录**”，在**“包含目录”**中添加API库的include目录，在**“库目录”**中添加API库的lib目录。



3. 配置外部依赖库。

   右键单击工程，单击**“属性”**，选择“**配置属性 -> 链接器 -> 输入**”，在**“附加依赖项”**中添加API库的lib。

   - 32位API库：EstunRemoteApiLib-vc140-x32.lib
   - 64位API库：EstunRemoteApiLib-vc140-x64.lib

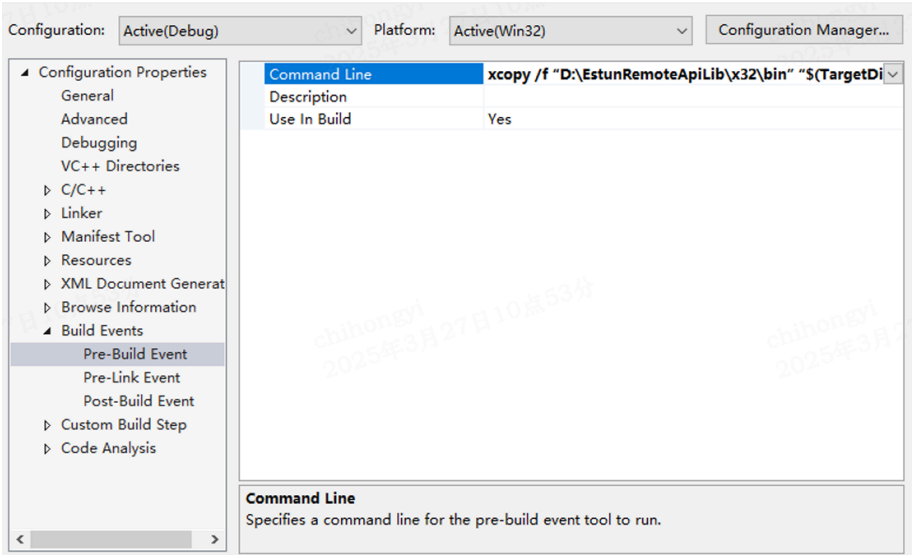


4. 配置生成事件。

   右键单击工程，单击**“属性”**，选择“**配置属性 -> 生成事件 -> 后期生成事件**”，在**“命令行”**中添加`xcopy /f "D:\EstunRemoteApiLib\x32\bin" "$(TargetDir)"`。

说明

D:\EstunRemoteApiLib请替换为自己安装API库的目录。



5. 添加程序逻辑，调用API接口。
   包含头文件：

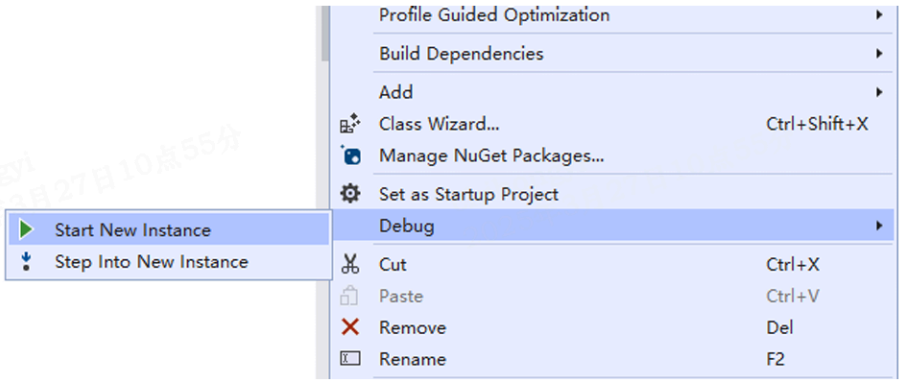
```
# include <EstunRemoteApiLib/EstunRemoteApiLib.h>
```

   实现调用逻辑：

```
//Create API library object
EstunRemoteApiLib*api = new EstunRemoteApiLib;

//Connect to robot server
api->connectToServer("127.0.0.1");

//Request motion control permission
E_ROB_PERMIT permit =api->E_AcquirePermit();

//Enable robot excitation
api->E_SetMot(true);

//Excute MovJ command
E_ROB_JOBID jobId = api->E_MovJ("t_g.P0");
```

```
//Query current robot joint position
E_ROB_POS pos = api->E_GetCurJPos();

//Release permission
api->E_ReleasePermit(permit);

//Disconnect
api->disConnect();

//Release API library object
delete api;
```

6. 编译、运行项目。
   右键单击项目，选择“**调试 -> 启动新实例**”启动实例，如果提示找不到xxx.dll，参考xxxxx



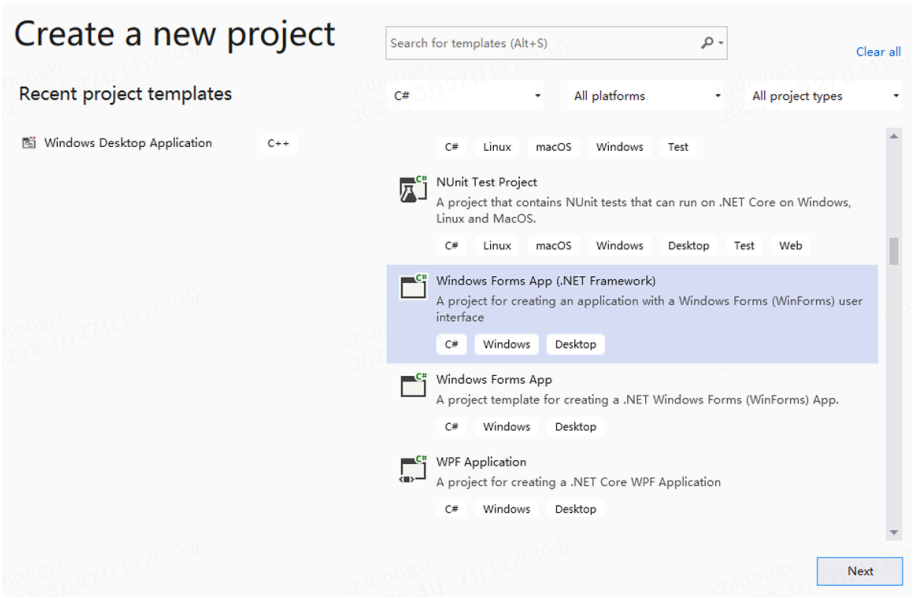
3.1.3 C#工程搭建

3.1.3.1 环境配置要求

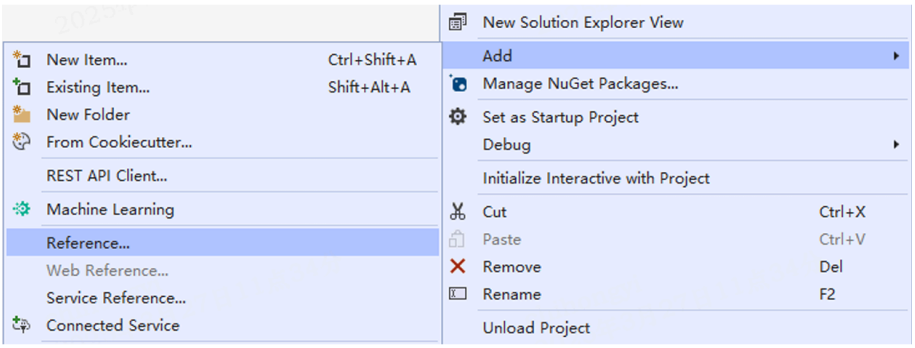
配置项	最低要求	推荐配置
系统版本	Win7	Win10
内存大小	4GB	8GB
VisualStudio版本	2015	2017
.Net版本	4.0	4.6

3.1.3.2 操作步骤

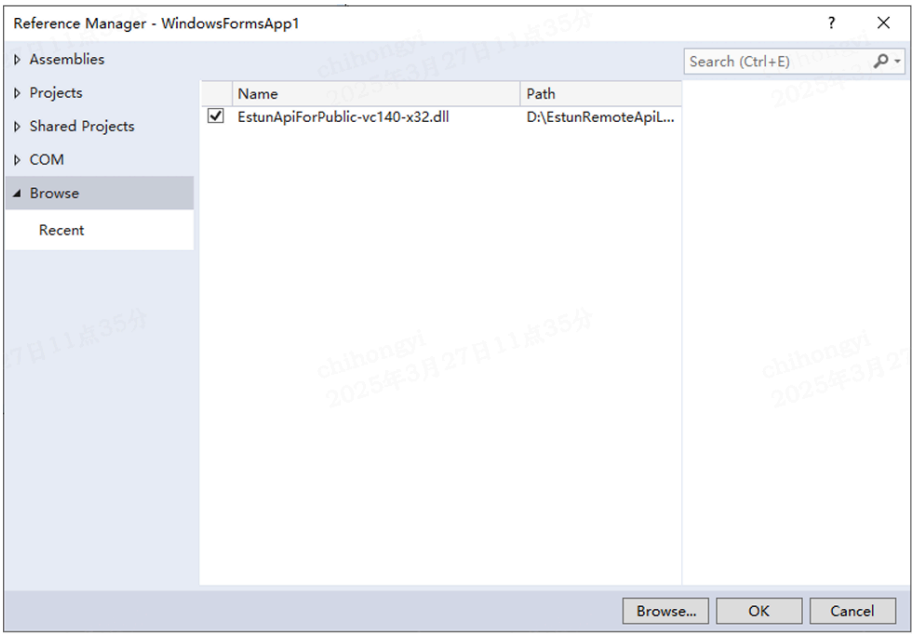
1. 新建工程。
   打开vs2015，单击左上角“**文件 -> 新建 -> 项目**”，选择“**Visual C# -> Windows窗体应用**”。



2. 添加引用。右键单击工程，选择“**添加 -> 引用**”。



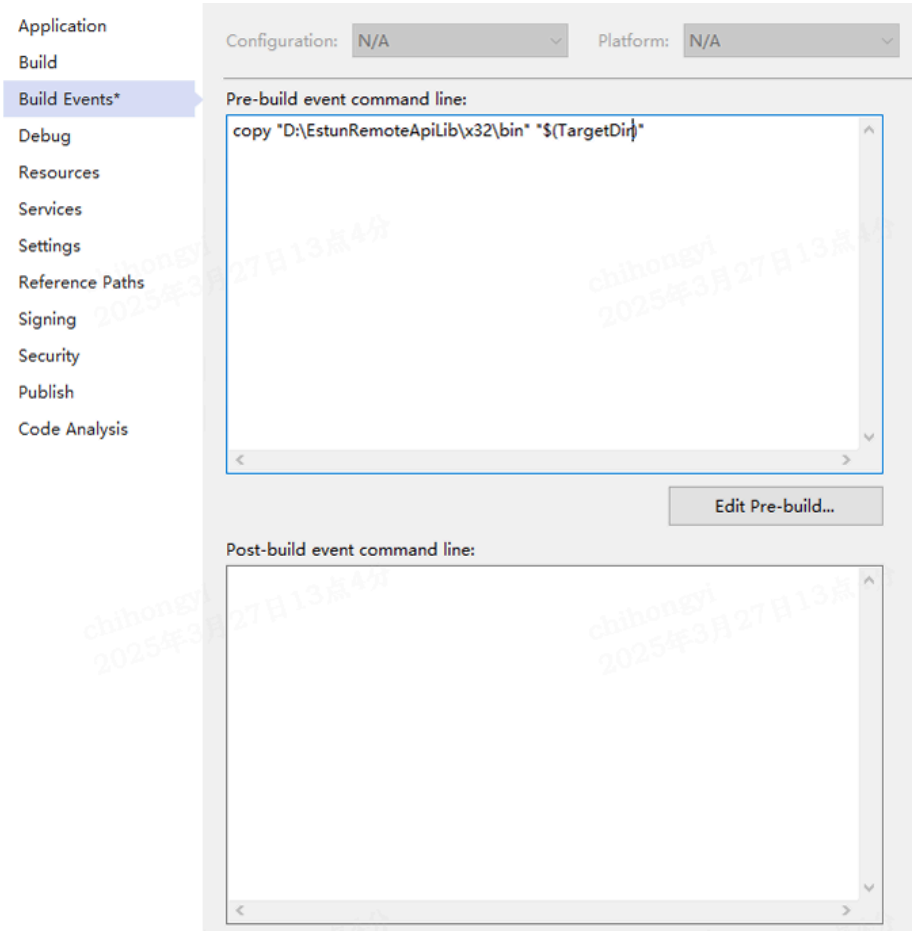
3. 单击右下角“**浏览**”，添加API库的依赖。此处以32位项目举例，添加`EstunRemote-ApiLib\x32\bin\ EstunApiFroPublic-vc140-x32.dll`，单击“**确定**”。



4. 配置生成事件。
   右键单击工程，单击**“属性”**，选择**“生成事件”**，在**“命令行”**中添加`copy "D:\EstunRemoteApiLib\x32\bin" "$(TargetDir)"`。

说明

`D:\EstunRemoteApiLib`请替换为自己安装API库的目录。

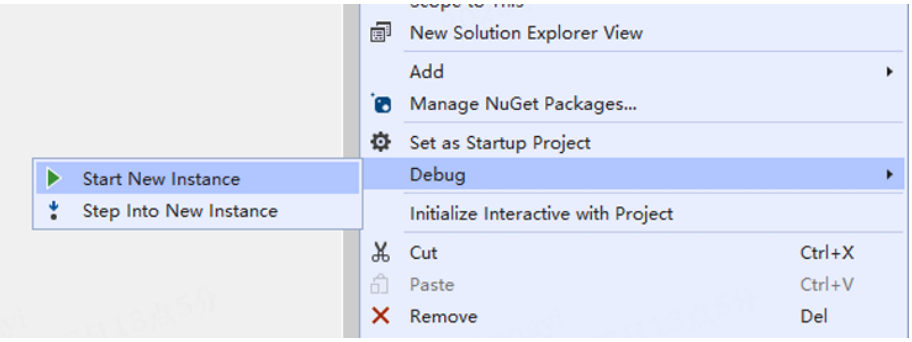


5. 添加程序逻辑，调用API接口。
   实现调用逻辑：

```
//Create API library object
EstunApiFroPublic.EstunApi_CLI api = new EstunApiFroPublic.EstunApi_CLI();

//Connect to robot server
api.connectToServer("127.0.0.1",true);

//Request motion control permission
EstunApiStruct_CLI.E_ROB_PERMIT_CLI permit = api.E_AcquirePermit();

//Enable robot excitation
api.E_SetMot(true);

//Excute MovJ command
EstunApiStruct_CLI.E_ROB_JOBID_CLI jobId = api.E_MovJ("t_g.P0");
```

```
//Query current robot joint position
EstunApiStruct_CLI.E_ROB_POS_CLI pos = api.E_GetCurJPos();

//Release permission
api.E_ReleasePermit(permit);

//Disconnect
api.disConnect();

//Release API library object
api.Dispose();
```

6. 编译、运行项目。
   右键单击项目，选择“**调试 -> 启动新实力**”启动实例，如果提示找不到xxx.dll，参考xxxxx



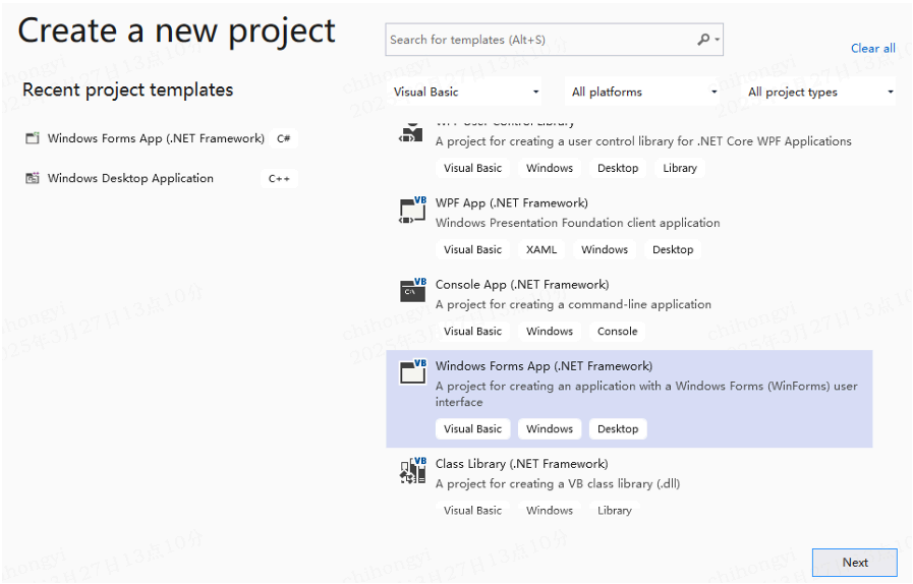
3.2 VB工程搭建

3.2.1 环境配置要求

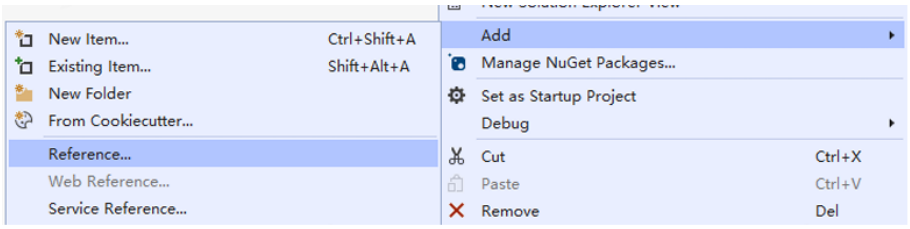
配置项	最低要求	推荐配置
系统版本	Win7	Win10
内存大小	4GB	8GB
VisualStudio版本	2015	2017
.Net版本	4.0	4.6

3.2.2 操作步骤

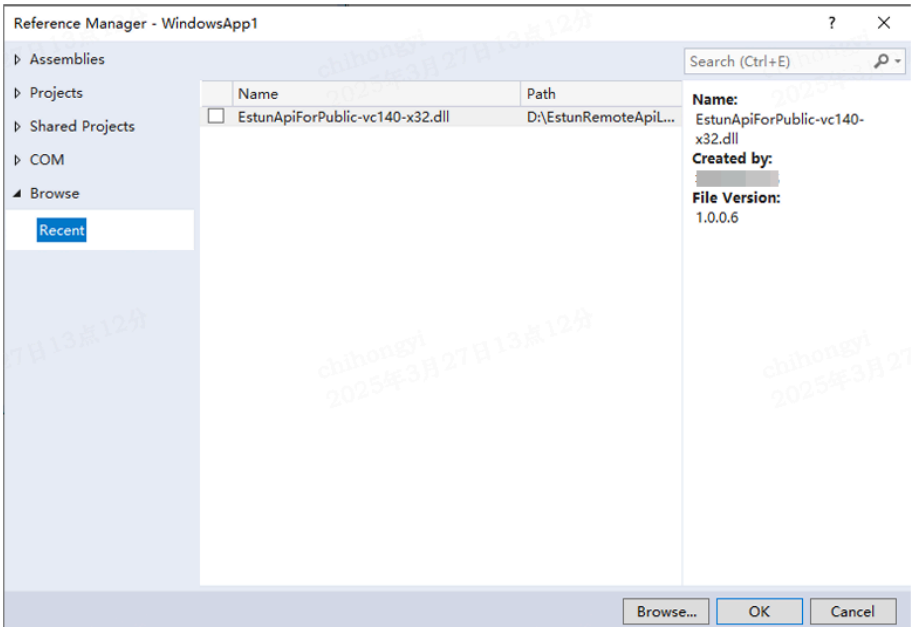
1. 新建工程。
   打开vs2015，单击左上角“**文件 -> 新建 -> 项目**”，选择“**Visual Basic -> Windows窗体应用**”。



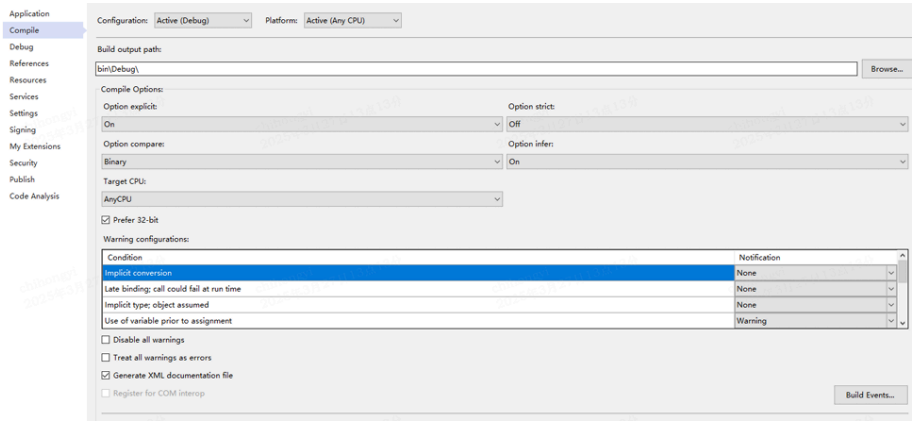
2. 添加引用。右键单击工程，选择“**添加 -> 引用**”。



3. 单击右下角**“浏览”**，添加API库的依赖。此处以32位项目举例，添加`EstunRemoteApiLib\x32\bin\ EstunApiFroPublic-vc140-x32.dll`，单击**“确定”**。



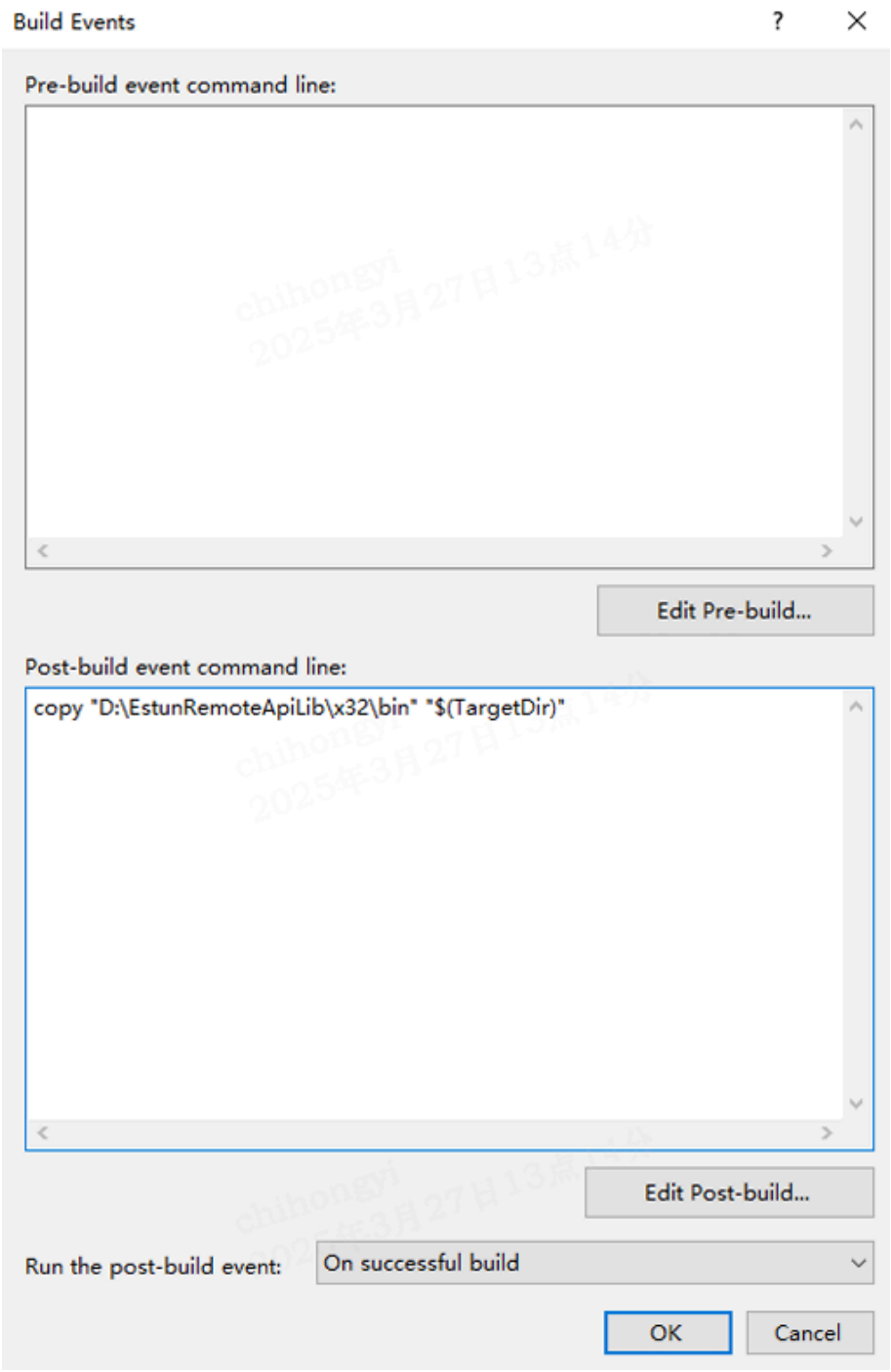
4. 配置生成事件。
   右键单击工程，单击**“属性”**，选择**“生成事件”**。



5. 在**“后期生成事件命令行”**中添加`copy "D:\EstunRemoteApiLib\x32\bin" "$(TargetDir)"`。

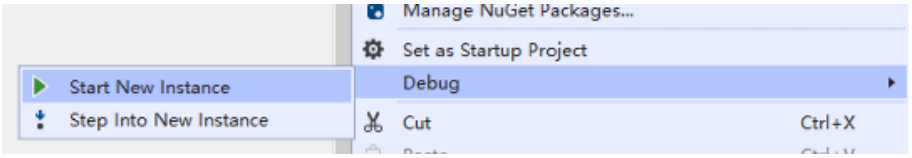
说明

`D:\EstunRemoteApiLib`请替换为自己安装API库的目录。



6. 添加程序逻辑，调用API接口。
   实现调用逻辑。
7. 编译、运行项目。

右键单击项目，选择“**调试 -> 启动新实力**”启动实例，如果提示找不到xxx.dll，参考xxxxx



4 运行环境

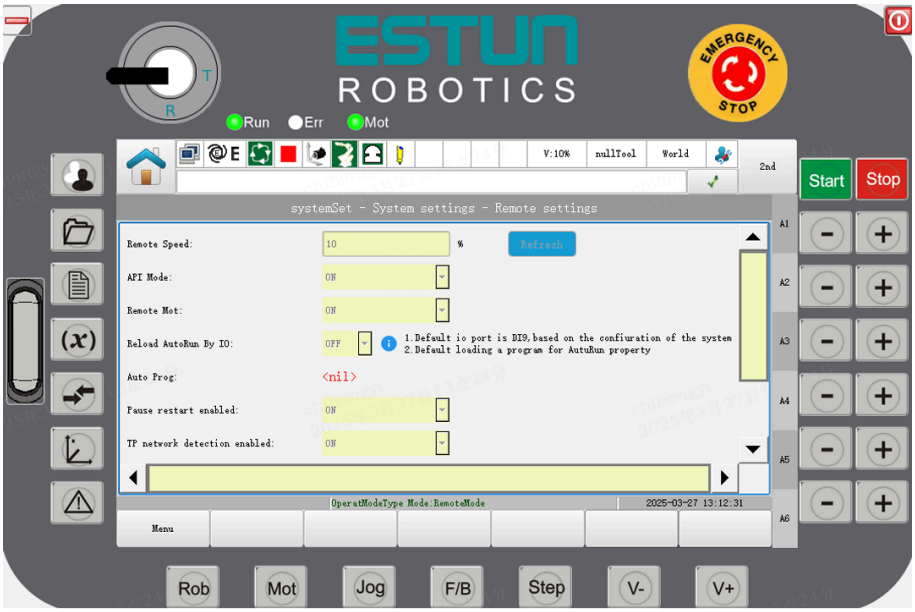
介绍基于API库开发的上位机程序与实际机器人的连接使用。

4.1 软件支持

- 控制系统为RCS2 V1.28.00 及以上版本。
- 目标控制器上需运行APIServer.exe程序。

4.2 示教器端配置

- 使用API功能控制机器人运动时，请将使能API Mode，并将示教器上的模式开关切换至远程模式。



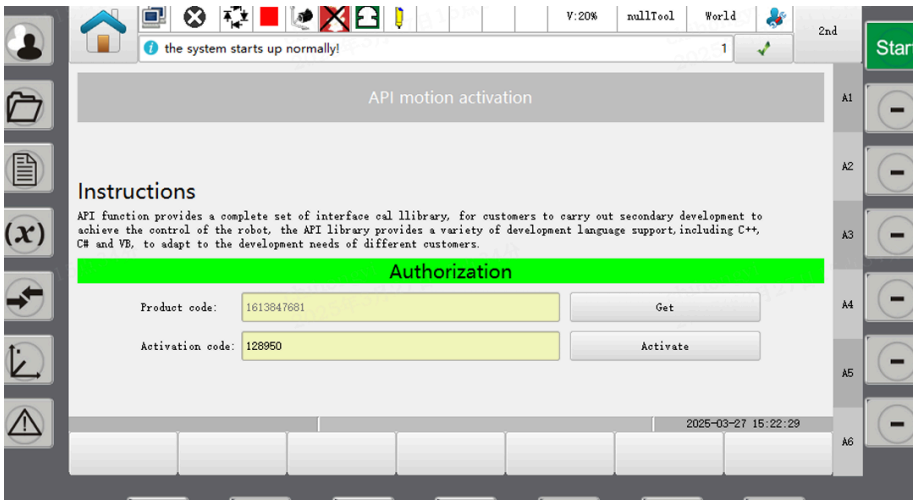
- 使用API功能的运动接口时，请授权激活API 功能。可通过以下步骤进行激活：

1. 在管理员权限下进入用户设置的API激活界面，通过点击界面中**“获取”**获取产品码，将该产品码发送给厂家，通过厂家获取激活码（激活码与产品码一一对应）。

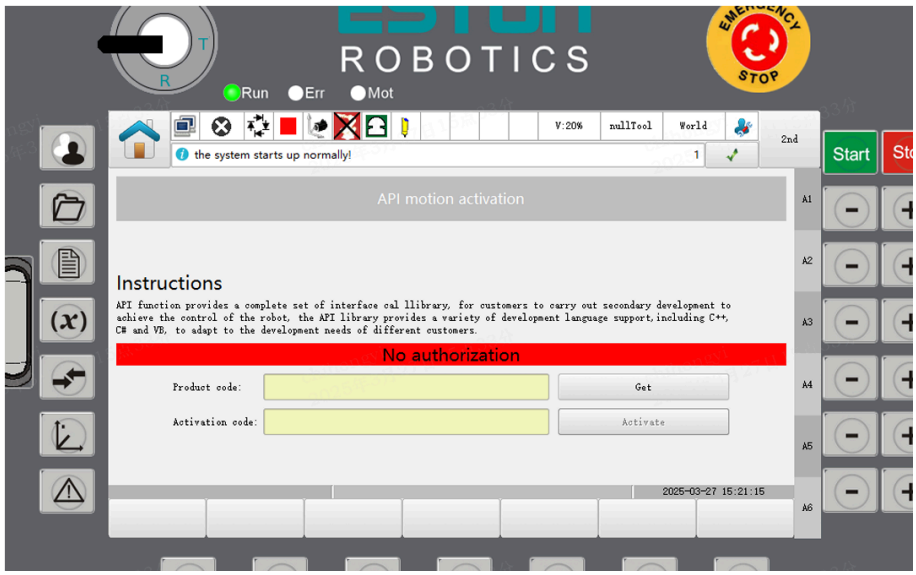
> **说明**
>
> 若进入界面时，激活状态为已激活，则不需要此激活配置过程。

2. 将获取的激活码输入到激活码编辑框中，点击**“激活”**。

- 若输入的激活码正确，则状态变为授权。



- 若输入的激活码不正确，则状态仍为未授权。



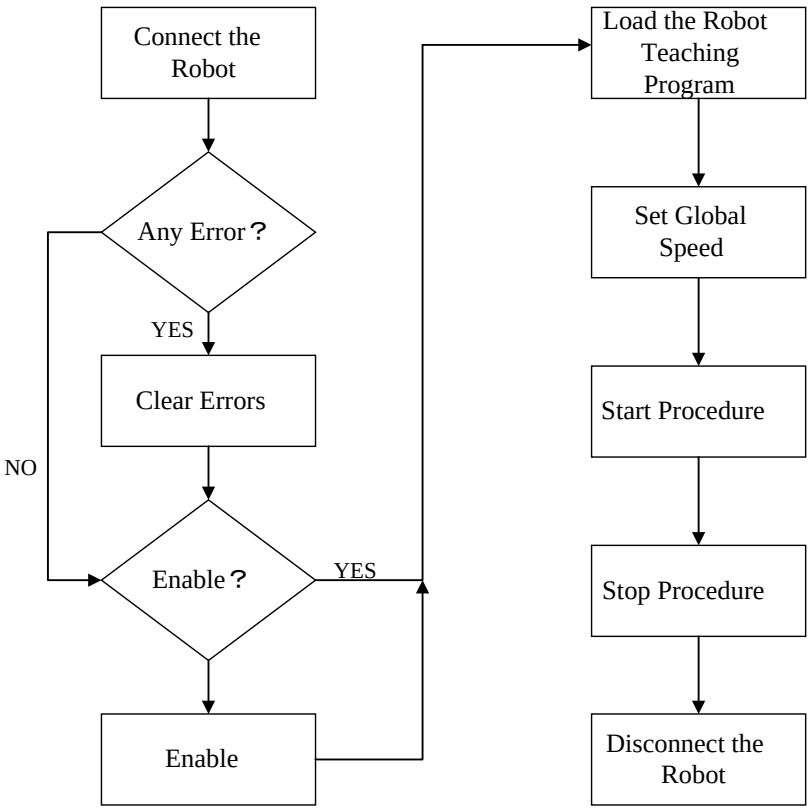
5 典型应用案例

通过典型案例介绍API接口的调用过程。

5.1 上位机加载机器人示教程序并再现运行

在进行该操作前，请确认目标机器人的示教程序可以正常运行，同时将示教器上的模式开关切换至再现模式。

整体流程：



示例代码如下：

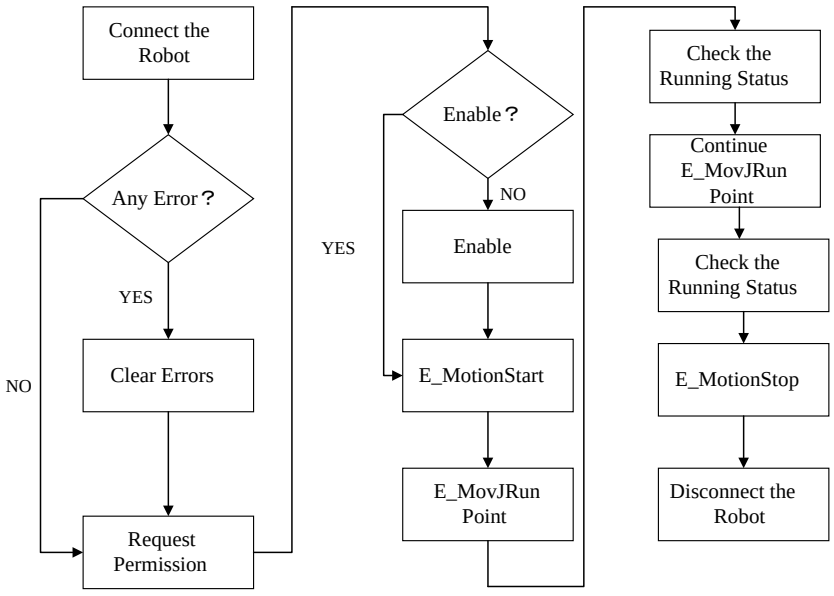
```
bool ret = false;
EstunRemoteApiLib *api = new EstunRemoteApiLib;
ret = api->connectToServer("192.168.6.63");
if (!ret)
{
   //Add exception handling code here
   return;
}
if (0 != api->E  GetErrorId())
{

   api->E_ClearError();
}
```

```
ret = api->E_SetServoOn(true);
if (!ret)
{
   //Add exception handling code here
   return;
}
ret = api->E_LoadProgame("Estun","test");
if (!ret)
{
   //Add exception handling code here
   return;
}
ret = api->E_SetGlobalSpeed(20);
if (!ret)
{
   //Add exception handling code here
   return;
}
ret = api->E_ProgRun();
if (!ret)
{
   //Add exception handling code here
   return;
}
ret = api->E_ProgStop();
if (!ret)
{
   //Add exception handling code here
   return;
}
api->disConnect();
delete api;
```

5.2 上位机规划点位控制机器人运动

在进行该操作前，请确认目标机器人可以安全到达上位机规划的点位，同时将示教器上的模式开关切换至远程模式，并使能API Mode。

整体流程：



示例代码如下：

```
bool ret = false;
EstunRemoteApiLib *api = new EstunRemoteApiLib;

ret = api->connectToServer("192.168.6.63");
if (!ret)
{
   //Add exception handling code here
   return;
}
if (0 != api->E_GetErrorId())
{

   api->E_ClearError();
}
E_ROB_PERMIT permit = api->E_AcquirePermit();
if (permit.m_mainctrlcode <= 1 )
{
   //Add exception handling code here
   return;
}
ret = api->E_SetServoOn(true);
if (!ret)
{
   //Add exception handling code here
   return;
}
ret = api->E_MotionStart();
if (!ret)
{
   //Add exception handling code here
   return;
}
E_ROB_POS pos = E_ROB_POS(0,0,0,0,30,0);
E_ROB_JOBID jobid = api->E_MovJ2(pos);
int status = api->E  GetCurMovCmdSts();
if (status == 0)
{
   //Exercise incomplete, custom operations, support cyclic query
}
std::vector<double> tmpApos;
tmpApos.clear();
tmpApos.push_back(90);
tmpApos.push_back(0);
tmpApos.push  back(0);
tmpApos.push_back(0);
tmpApos.push_back(90);
tmpApos.push  back(0);
pos.setAPosByArray(tmpApos);
jobid = api->E_MovJ2(pos);
status = api->E  GetMovCmdSts(jobid);
if (status == 0)
{
   //Exercise incomplete, custom operations, support cyclic query
}
ret = api->E_MotionStop();
if (!ret)
{
   //Add exception handling code here
return;
}
api->disConnect();
delete api;
```

6 接口参考

详细介绍API库提供的接口功能以及相关参数说明。

6.1 结构化数据

6.1.1 内部字符串

功能介绍

Class EstunRobot::ESString，API 库内置的字符串数据类型，用于承载与机器人交互过程中的字符串相关数据。

参数说明

无

配置示例

- 默认构造函数，构造一个空字符串。

```
ESString ()
```

- 构造函数，构造一个内容为c_str的字符串。

```
ESString (const char *  c_str)
```

- 拷贝构造函数。

```
ESString (const ESString &  t)
```

- 析构函数。

```
~ESString()
```

- 赋值运算符: 将 const ESString 对象的值拷贝到指定对象。

```
ESString& operator=(const ESString& t)
```

- 赋值运算符: 将 const char* 对象的值拷贝到指定对象。

```
ESString& operator=(const char* t)
```

- 加法运算符: 将 const ESString 对象的值追加到指定对象。

```
ESString& operator+(const ESString& t)
```

- 加法运算符: 将 const char* 对象的值追加到指定对象。

```
ESString& operator+(const char* t)
```

- 关系运算符：判断两个给定对象是否相等。

```
bool operator==(const char* t)
```

- 获取字符串大小。

```
size_t size()
```

- 获取字符串内容。

```
 const char* c_str() const
```

6.1.2 权限信息

功能介绍

Struct EstunRobot::E_ROB_PERMIT，用于表示操作机器人所需的权限信息。

参数说明

- m_mainctrlcode：权限唯一码，数据类型为uint16_t。大于1的整数表示有运动操作权限，否则没有该权限。
- m_timestamp：时间戳，数据类型为uint16_t。时间戳大于0表示正确获取到权限，否则获取权限失败。

6.1.3 坐标类型

功能介绍

enum E_CoordType ，用于定义机器人运动控制中涉及的坐标类型，不同枚举值对应不同的坐标体系，便于程序中统一标识和处理各类坐标相关逻辑。

参数说明

- axisCoord = 0：关节坐标，标识机器人以关节角度为基准的坐标体系。
- worldCoord：世界坐标，标识机器人基于全局固定坐标系的坐标体系（默认值为 1）。

- toolCoord：工具坐标，标识以机器人末端工具中心点为基准的坐标体系（默认值为 2）。
- usrCoord：用户坐标，标识用户自定义的局部坐标系（默认值为 3）。
- errCoord：错误坐标，标识坐标类型无效或识别出错（默认值为 4）。

6.1.4 系统模式

功能介绍

enum E_SysModeType ，用于定义机器人系统的运行模式类型，不同枚举值对应不同的系统控制模式，便于程序中统一标识和处理各类系统模式相关逻辑。

参数说明

- manualMode = 0：手动模式，标识机器人处于人工手动操作的运行模式。
- autoMode：自动模式，标识机器人处于自主自动化运行的模式（默认值为 1）。
- remoteMode：远程模式，标识机器人处于远程控制的运行模式（默认值为 2）。
- banMode：禁止模式，标识机器人被禁止运行的模式（默认值为 3）。
- mutil：多工位模式，标识机器人处于多工位协同运行的模式（默认值为 4）。
- API：API 模式，标识机器人通过 API 接口被控制的运行模式（默认值为 5）。
- errSysMode：错误模式，标识机器人系统模式无效或识别出错（默认值为 6）。

6.1.5 运动模式

功能介绍

enum E_RunModeType ，用于定义机器人的运动运行模式类型，不同枚举值对应不同的运动控制模式，便于程序中统一标识和处理各类运动模式相关逻辑。

参数说明

- step = 1：单步模式，标识机器人以单步执行的方式进行运动。
- sustain：连续模式，标识机器人以连续运行的方式进行运动（默认值为 2）。
- errRunMode：错误模式，标识机器人运动模式无效或识别出错（默认值为 3）。

6.1.6 IO类型

功能介绍

enum E_IOType ，用于定义机器人的 IO 端口类型，不同枚举值对应不同的 IO 信号类型，便于程序中统一标识和处理各类 IO 端口相关逻辑。

参数说明

- DI = 0：数字输入端口，标识机器人的数字量输入 IO 类型。
- DO = 1：数字输出端口，标识机器人的数字量输出 IO 类型。
- AO = 2：模拟输出端口，标识机器人的模拟量输出 IO 类型。
- AI = 3：模拟输入端口，标识机器人的模拟量输入 IO 类型。
- SIM_DI = 11：仿真数字输入端口，标识机器人的仿真数字量输入 IO 类型。
- SIM_DO = 12：仿真数字输出端口，标识机器人的仿真数字量输出 IO 类型。
- SIM_AI = 13：仿真模拟输入端口，标识机器人的仿真模拟量输入 IO 类型。
- SIM_AO = 14：仿真模拟输出端口，标识机器人的仿真模拟量输出 IO 类型。
- SIM_GI = 15：仿真通用输入端口，标识机器人的仿真通用型输入 IO 类型（默认值为 15）。
- SIM_GO = 16：仿真通用输出端口，标识机器人的仿真通用型输出 IO 类型（默认值为 16）。

6.1.7 变量作用域类型

功能介绍

enum E_ScopeType ，用于定义机器人程序中变量的作用域类型，不同枚举值对应不同的变量生效范围，便于程序中统一标识和管理各类变量的作用域。

参数说明

- system = 0：系统域，标识变量作用于整个机器人系统层面，全局生效且不受工程 / 程序影响。
- global：全局域，标识变量作用于机器人全局范围（默认值为 1），跨工程生效。
- project：工程域，标识变量仅作用于当前工程范围内（默认值为 2）。
- local：程序本地域，标识变量仅作用于所属程序内部（默认值为 3），为局部变量。

6.1.8 用户坐标系类型

功能介绍

enum E_UserCoordType， 用于定义机器人用户坐标系的类型，不同枚举值对应不同的坐标系类别，便于程序中统一标识和处理各类用户坐标系相关逻辑。

参数说明

- ErrorUserCoord = -1：类型错误，标识用户坐标系类型无效或识别出错。
- UsrCoord = 0：用户坐标，标识机器人自定义的用户坐标系。
- ExtCoord = 1：外部坐标系，标识机器人外接设备对应的坐标系。
- SYNCoord = 3：同步坐标系，标识机器人用于同步控制的坐标系（默认值为 3）。

6.1.9 报警等级

功能介绍

enum E_ErrorLevel ，用于定义机器人系统的报警等级类型，不同枚举值对应不同严重程度的告警信息，便于程序中统一标识和处理各类告警相关逻辑。

参数说明

- fault_error = 1：严重错误，标识机器人系统发生会影响运行的严重故障类告警。
- error：普通错误，标识机器人系统发生的一般性错误类告警（默认值为 2）。
- warning：警告，标识机器人系统存在潜在风险的告警提示（默认值为 3）。
- info：提示，标识机器人系统的常规状态类提示信息（默认值为 4）。

6.1.10 运动状态

功能介绍

enum E_RunStatusType ，用于定义机器人的运动状态类型，不同枚举值对应不同的运动状态，便于程序中统一标识和处理各类运动状态相关逻辑。

参数说明

- running = 1：运行，标识机器人处于正常运动运行的状态。
- suspend：暂停，标识机器人运动过程中被暂停的状态（默认值为 2）。
- stop：停止，标识机器人完全停止运动的状态（默认值为 3）。
- errRunStatus：错误，标识机器人运动状态异常或识别出错（默认值为 4）。

6.1.11 使能状态

功能介绍

enum E_ServoStatusType ，用于定义机器人伺服电机的使能状态类型，不同枚举值对应不同的伺服使能状态，便于程序中统一标识和处理各类伺服状态相关逻辑。

参数说明

- ServoOff = 0：未使能，标识机器人伺服电机处于未启用的状态。
- ServoOn：使能，标识机器人伺服电机处于启用的状态（默认值为 1）。
- errStatus：错误，标识机器人伺服电机状态异常或识别出错（默认值为 3）。

6.1.12 安装方式

功能介绍

enum E_RobotInstallType，用于定义机器人的安装方式类型，不同枚举值对应不同的安装形态，便于程序中统一标识和处理各类安装方式相关逻辑。

参数说明

- Normal = 0：标准安装，标识机器人采用常规的落地 / 基座式安装方式（注释标注值为 1，枚举定义值为 0，以枚举定义为准）。
- Ceiling：吊装，标识机器人采用倒挂式安装方式（默认值为 1，注释标注值为 2，以枚举定义为准）。
- ErrType：错误类型，标识机器人安装方式识别出错或无效（默认值为 2）。

6.1.13 震动抑制等级

功能介绍

enum E_SuppresLevel ，用于定义机器人的震动抑制等级类型，不同枚举值对应不同强度的震动抑制策略，便于程序中统一标识和处理各类震动抑制相关逻辑。

参数说明

- Standard = 0：标准等级，标识机器人采用默认的基础震动抑制策略。
- High：高等级，标识机器人采用高强度的震动抑制策略（默认值为 1）。
- MidHigh：中高等级，标识机器人采用中高强度的震动抑制策略（默认值为 2）。
- MidLow：中低等级，标识机器人采用中低强度的震动抑制策略（默认值为 3）。

- Low：低等级，标识机器人采用低强度的震动抑制策略（默认值为 4）。
- ErrLevel：错误等级，标识机器人震动抑制等级识别出错或无效（默认值为 5）。

6.1.14 指令标识

功能介绍

struct E_ROB_JOBID，用于定义机器人运动指令的标识信息，包含指令 ID 和时间戳，用于标识指令下发状态及下发时间。

参数说明

- m_jobID：指令 ID，数据类型为 int；大于 0 表示指令下发成功，否则表示指令下发失败，构造函数中默认初始值为 - 1。
- m_timeStamp：时间戳，数据类型为 uint32_t；记录运动指令下发成功时的当前时间戳，构造函数中默认初始值为 0。

6.1.15 工程程序名

功能介绍

struct E_ROB_PROGNAME ，用于定义机器人工程和程序的名称信息，包含当前加载、运行的工程名与程序名，便于标识机器人当前的程序运行上下文。

参数说明

- prjname：当前加载的工程名，数据类型为 ESString。
- programname：当前加载的程序名，数据类型为 ESString。
- exeprogramName：当前运行的程序名，数据类型为 ESString；若运行的工程非当前加载工程，格式为 "工程名;程序名"。

6.1.16 附加指令

功能介绍

struct E_ROB_APIBUFF，用于定义机器人的附加指令信息，存储指令类型和二进制形式的指令内容，需调用 AddDoTypeConvert 相关接口转换结构参数得到具体内容。

参数说明

- m_type：附加指令类型，数据类型为 uint8_t，构造函数中默认初始值为 0。
- m_data：指令内容二进制，数据类型为 uint8_t 数组（长度为 APIBUFFLENGTH），构造函数中通过 memset 将数组内存置 0。

6.1.17 附加指令结构一

功能介绍

struct E_ROB_ADDDO_I，用于定义机器人附加指令中与 IO 相关的参数信息，包含 IO 类型、序号、值及触发相关配置等。

参数说明

- ioType：IO 类型，数据类型为 E_IOType，构造函数中默认初始值为 DI。
- ioindex：IO 序号，数据类型为 int，构造函数中默认初始值为 0。
- iovalue：IO 值，数据类型为 double，构造函数中默认初始值为 0。
- trigType：触发类型，数据类型为 int；1 表示路径百分比触发，2 表示距离触发，构造函数中默认初始值为 0。
- exTimes：触发延时，数据类型为 int，单位毫秒，负数为提前触发，构造函数中默认初始值为 0。
- exPercent：当前运动执行触发百分比，数据类型为 int，构造函数中默认初始值为 0。
- exDistance：当前运动执行触发距离，数据类型为 double，构造函数中默认初始值为 0。

6.1.18 空间坐标形态配置数据

功能介绍

class E_ROB_POSCFG ，继承自 IObject，用于定义机器人空间位置的形态配置数据，主要管理关节位置模式和各关节轴角度的象限值，提供构造、赋值、取值等一系列操作接口。

配置说明

- 默认构造函数，初始化后值无效。

```
E_ROB_POSCFG ()
```

- 基本构造函数；mode 为关节位置模式（取值范围 [0,7]），cf1~cf6 分别为 1~6 号关节轴角度所在的象限值，默认值均为 0。

```
E_ROB_POSCFG (int mode, int cf1 = 0, int cf2 = 0, int cf3 = 0, int cf4 = 0, int cf5 = 0,
 int cf6 = 0)
```

- 数组参数构造函数；c_array 为形态配置数据数组，格式为 {mode,cf1,cf2,cf3,cf4,cf5,cf6}，debug 模式下不建议使用。

```
E_ROB_POSCFG(const std::vector<int>& c_array)
```

- 数组参数构造函数；c_array 为形态配置数据数组，格式为 {mode,cf1,cf2,cf3,cf4,cf5,cf6}，debug 模式下不建议使用。

```
E_ROB_POSCFG (int c_array [7])
```

- 拷贝构造函数；t 为空间位置形态配置数据源对象。

```
E_ROB_POSCFG (const E_ROB_POSCFG& t)
```

成员函数

- 析构函数，释放类内资源。

```
~E_ROB_POSCFG ()
```

- 设置形态配置数据；array 格式为 {mode,cf1,cf2,cf3,cf4,cf5,cf6}，debug 模式下不可以使用。

```
setCfgByArray(const std::vector<int>& array) /setCfgByArray (int array [7])
```

- 获取关节位置模式，返回值为 int 类型，取值范围 [0,7]。

```
getMode () const
```

- 设置关节位置模式；val 为关节位置模式值，取值范围 [0,7]。

```
setMode (int val)
```

- 获取指定轴角度的象限值；index 为轴号（对应 mode、cf1~cf6），返回值为该轴角度的象限值。

```
getCF_ValByIndex (int index) const
```

- 设置指定轴角度的象限值；index 为轴号（对应 mode、cf1~cf6），val 为该轴角度的象限值。

```
setCF_ValByIndex (int index, int val)
```

- 获取空间位置形态配置数据数组；返回 / 输出格式为 {mode,cf1,cf2,cf3,cf4,cf5,cf6} 的数组，debug 模式下不建议使用。

```
getValueList () const /getValueList (int res [7])
```

6.1.19 机器人位置

功能介绍

class E_ROB_POS，用于定义机器人的位置信息，支持轴关节坐标和空间坐标两种类型的存储与操作，提供构造、赋值、坐标类型切换、各维度坐标读写等完整接口。

参数说明

无

配置说明

- 默认构造函数，初始化后值无效。

```
E_ROB_POS ()
```

- 轴关节坐标构造函数；j1~j16 分别为 1~16 号关节轴的坐标，其中 j7~j16 为选填项，默认值均为 0。

```
E_ROB_POS (double j1, double j2, double j3, double j4, double j5, double j6,
 double j7 = 0, double j8 = 0, double j9 = 0, double j10 = 0, double j11 = 0, double
 j12 = 0, double j13 = 0, double j14 = 0, double j15 = 0, double j16 = 0)
```

- 轴关节坐标数组构造函数；apos_Array 为坐标数组，格式为 {j1,j2,j3...j16}，debug 模式下不可以使用。

```
E_ROB_POS(const std::vector<double>& apos_Array)
```

- 轴关节坐标数组构造函数；apos_Array 为坐标数组，格式为 {j1,j2,j3...j16}，debug 模式下不可以议使用。

```
E_ROB_POS (double apos_Array [16])
```

- 空间坐标构造函数；cfg 为空间位置形态配置数据，x/y/z 为 TCP 点在参考坐标系上对应方向的坐标，a/b/c 为 TCP 点相对参考坐标系 z/y/x 轴旋转的欧拉角，a7~a16 为附加轴 7~16 号轴的坐标（选填，默认值为 0）。

```
E_ROB_POS (const E_ROB_POSCFG& cfg, double x, double y, double z, double a,
 double b, double c, double a7 = 0, double a8 = 0, double a9 = 0, double a10 = 0,
 double a11 = 0, double a12 = 0, double a13 = 0, double a14 = 0, double a15 = 0,
 double a16 = 0)
```

- 空间坐标数组构造函数；cfg_Array 为形态配置数据（{mode,cf1,cf2,cf3,cf4,cf5,cf6}），cpos_Array 为空间坐标数组（{x,y,z,a,b,c,a7...a16}），debug 模式下不建议使用。

```
E_ROB_POS(const std::vector<int>& cfg_Array, const std::vector<double>& cpos_Array) / E_ROB_POS (int cfg_Array [7], double cpos_Array [16])
```

- 拷贝构造函数；t 为轴关节坐标 / 空间坐标数据源对象。

```
E_ROB_POS (const E_ROB_POS& t)
```

成员函数

- 析构函数，释放类内资源。

```
~E_ROB_POS ()
```

- 赋值运算符重载；t 为轴关节坐标 / 空间坐标数据源对象，返回当前对象的引用。

```
E_ROB_POS& operator=(const E_ROB_POS& t)
```

- 坐标类型枚举；PT_APos = 0 表示轴关节坐标，PT_CPos 表示空间坐标（默认值为 1）。

```
PosType
```

- 获取坐标类型，返回值为 PosType 枚举类型。

```
getPosType () const
```

- 设置坐标类型；pt 为目标坐标类型（PT_APos/PT_CPos）。

```
setPosType (PosType pt)
```

- 设置轴关节坐标数组；array 格式为 {j1,j2,j3...j16}，debug 模式下不建议使用。

```
setAPosByArray(const std::vector<double>& array) /setAPosByArray (double array [16])
```

- 获取指定轴的关节坐标；index 为轴号（取值范围 [1,16]），返回值为该轴的坐标值。

```
getApos_ValByIndex (int index) const
```

- 设置指定轴的关节坐标；index 为轴号（取值范围 [1,16]），val 为目标坐标值。

```
setApos_ValByIndex (int index, double val)
```

- 设置空间位置形态配置数据；array 格式为 {mode,cf1,cf2,cf3,cf4,cf5,cf6}，debug 模式下不建议使用。

```
setCfgByArray(const std::vector<double>& array) /setCfgByArray (int array [7])
```

- 获取 Cfg 值；outData 为输出参数，存储获取到的 cfg 数据，返回 bool 值（true 成功，false 失败）。

```
GetCfgValue (E_ROB_POSCFG& outData)
```

- 设置 Cfg 值；dataIn 为待设置的 cfg 数据源对象。

```
SetCfgValue (E_ROB_POSCFG dataIn)
```

- 设置空间坐标数组；array 格式为 {x,y,z,a,b,c,a7...a16}，debug 模式下不建议使用。

```
setCPosByArray(const std::vector<double>& array) /setCPosByArray (double array [16])
```

- 分别获取 TCP 点在参考坐标系 x/y/z 方向的坐标，返回值为 double 类型。

```
getX ()/getY ()/getZ () const
```

- 分别设置 TCP 点在参考坐标系 x/y/z 方向的坐标；val 为目标坐标值。

```
setX (double val)/setY (double val)/setZ (double val)
```

- 分别获取 TCP 点相对参考坐标系 z/y/x 轴旋转的欧拉角，返回值为 double 类型。

```
getA ()/getB ()/getC () const
```

- 分别设置 TCP 点相对参考坐标系 z/y/x 轴旋转的欧拉角；val 为目标角度值。

```
setA (double val)/setB (double val)/setC (double val)
```

- 获取空间坐标的附加轴关节坐标；index 为轴号（取值范围 [7,16]），返回值为该附加轴的坐标值。

```
getExValByIndex (int index) const
```

- 设置空间坐标的附加轴关节坐标；index 为轴号（取值范围 [7,16]），val 为目标坐标值。

```
setExValByIndex (int index, double val)
```

- 获取位置的值数组；坐标类型为轴关节坐标时返回 {j1,j2,j3...j16}，为空间坐标时返回 {mode,cf1,cf2,cf3,cf4,cf5,cf6,...}，debug 模式下不建议使用。

```
getValueList () const /getValueList (double value [23])
```

6.1.20 机器人速度

功能介绍

class E_ROB_SPEED，用于定义机器人运动速度相关参数，涵盖关节速度百分比、TCP 线速度、空间旋转速度、外部轴线速度和外部轴角速度，提供参数的构造、赋值、读写等操作接口。

参数说明

无

配置说明

- 默认构造函数，初始化后值无效。

```
E_ROB_SPEED ()
```

- 基本构造函数；m_Per 为关节速度百分比，m_Tcp 为 TCP 线速度，m_ori 为空间旋转速度，m_exj_l 为外部轴线速度，m_exj_r 为外部轴角速度。

```
E_ROB_SPEED (double m_Per, double m_Tcp, double m_ori, double m_exj_l,
 double m_exj_r)
```

- 拷贝构造函数；t 为机器人运动速度数据源对象。

```
E_ROB_SPEED (const E_ROB_SPEED& t)
```

成员函数

- 析构函数，释放类内资源。

```
~E_ROB_SPEED ()
```

- 赋值运算符重载；t 为机器人运动速度数据源对象，返回当前对象的引用。

```
E_ROB_SPEED& operator=(const E_ROB_SPEED& t)
```

参数读写接口：

- get_Per () const：获取关节速度百分比，返回值为 double 类型。
- set_Per (double val)：设置关节速度百分比；val 为目标百分比值。
- get_Tcp () const：获取 TCP 线速度，返回值为 double 类型。
- set_Tcp (double val)：设置 TCP 线速度；val 为目标线速度值。
- get_ori () const：获取空间旋转速度，返回值为 double 类型。
- set_ori (double val)：设置空间旋转速度；val 为目标旋转速度值。
- get_exj_l () const：获取外部轴线速度，返回值为 double 类型。
- set_exj_l (double val)：设置外部轴线速度；val 为目标线速度值。
- get_exj_r () const：获取外部轴角速度，返回值为 double 类型。
- set_exj_r (double val)：设置外部轴角速度；val 为目标角速度值。

6.1.21 机器人过渡区域

功能介绍

class E_ROB_ZONE，用于定义机器人运动轨迹的转弯区参数，包含过渡类型、关节轨迹转弯百分比、笛卡尔空间转弯区大小、恒速过渡使能状态等核心配置，提供参数的构造、赋值、读写等操作接口。

参数说明

无

配置说明

- 默认构造函数，初始化后值无效。

```
E_ROB_ZONE ()
```

- 基本构造函数；m_ZoneType 为过渡类型（0 - 无过渡 / 1 - 绝对过渡 / 2 - 相对过渡），m_Per 为关节轨迹转弯百分比，m_dis 为笛卡尔空间转弯区大小，m_vConst 为恒速过渡使能状态。

```
E_ROB_ZONE (int8_t m_ZoneType, double m_Per, double m_dis, int m_vConst)
```

- 拷贝构造函数；t 为机器人转弯区数据源对象。

```
E_ROB_ZONE (const E_ROB_ZONE& t)
```

成员函数

- 析构函数，释放类内资源。

```
~E_ROB_ZONE ()
```

- 赋值运算符重载；t 为机器人转弯区数据源对象，返回当前对象的引用。

```
E_ROB_ZONE& operator=(const E_ROB_ZONE& t)
```

- 获取过渡类型，返回值为 int8_t 类型（0 - 无过渡，1 - 绝对过渡，2 - 相对过渡）。

```
get_ZoneType () const
```

- 设置过渡类型；val 为目标过渡类型（0 - 无过渡，1 - 绝对过渡，2 - 相对过渡）。

```
set_ZoneType (int8_t val)
```

- 获取关节轨迹转弯百分比，返回值为 double 类型。

```
get_Per () const
```

- 设置关节轨迹转弯百分比；val 为目标百分比值。

```
set_Per (double val)
```

- 获取笛卡尔空间转弯区大小，返回值为 double 类型。

```
get_Dis () const
```

- 设置笛卡尔空间转弯区大小；val 为目标大小值。

```
set_Dis (double val)
```

- 获取恒速过渡使能状态，返回值为 int 类型。

```
get_vConst () const
```

- 设置恒速过渡使能状态；val 为目标使能状态值。

```
set_vConst (int val)
```

6.1.22 负载参数

功能介绍

class E_ROB_PAYLOAD，用于定义机器人末端工具及负载的物理参数，涵盖重量、重心偏移量、各方向惯量矩 / 惯量积等核心属性，提供参数的构造、赋值、数组批量操作及单个参数读写等完整接口。

参数说明

无

返回值

- 默认构造函数，初始化后值无效。

```
E_ROB_PAYLOAD ()
```

- 基本构造函数；
  - m_M：工具 & 负载的重量（单位：kg）；
  - m_Mx/m_My/m_Mz：负载在 OTool-XYZ 坐标系 X/Y/Z 方向的偏移量（单位：mm）；
  - m_Ixx/m_Iyy/m_Izz：负载在重心位置 X/Y/Z 方向回转的惯量矩（单位：kg#mm²），默认值为 0；
  - m_Ixy/m_Ixz/m_Iyz：负载在重心位置 XY/XZ/YZ 交叉方向的惯量积（单位：kg#mm²），默认值为 0。

```
E_ROB_PAYLOAD (double m_M, double m_Mx, double m_My, double m_Mz,
 double m_Ixx = 0, double m_Iyy = 0, double m_Izz = 0, double m_Ixy = 0, double
 m_Ixz = 0, double m_Iyz = 0)
```

- 数组构造函数；array 为负载数据数组，格式为 {m_M,m_Mx,m_My,m_Mz,m_Ixx,m_-Iyy,m_Izz,m_Ixy,m_Ixz,m_Iyz}，debug 模式下不建议使用。

```
E_ROB_PAYLOAD(const std::vector<double>& array) / E_ROB_PAYLOAD (double
 array [10])
```

- 拷贝构造函数；t 为机器人负载数据源对象。

```
E_ROB_PAYLOAD (const E_ROB_PAYLOAD& t)
```

成员函数

- 析构函数，释放类内资源。

```
~E_ROB_PAYLOAD
```

- 赋值运算符重载；t 为机器人负载数据源对象，返回当前对象的引用。

```
E_ROB_PAYLOAD& operator=(const E_ROB_PAYLOAD& t)
```

- 批量设置负载参数；array 格式为 {m_M,m_Mx,m_My,m_Mz,m_Ixx,m_Iyy,m_Izz,m_Ixy,m_Ixz,m_Iyz}，debug 模式下不建议使用。

```
setPayloadByArray(const std::vector<double>& array) /setPayloadByArray
(double array [10])
```

- 获取工具 & 负载的重量（单位：kg），返回值为 double 类型。

```
get_M () const
```

- 设置工具 & 负载的重量；val 为目标重量值。

```
set_M (double val)
```

- 分别获取负载在 OTool-XYZ 坐标系 X/Y/Z 方向的偏移量（单位：mm），返回值为 double 类型。

```
get_Mx ()/get_My ()/get_Mz () const
```

- 分别设置负载在 OTool-XYZ 坐标系 X/Y/Z 方向的偏移量；val 为目标偏移量值。

```
set_Mx (double val)/set_My (double val)/set_Mz (double val)
```

- 分别获取负载在重心位置 X/Y/Z 方向回转的惯量矩（单位：kg#mm²），返回值为 double 类型。

```
get_Ixx ()/get_Iyy ()/get_Izz () const
```

- 分别设置负载在重心位置 X/Y/Z 方向回转的惯量矩；val 为目标惯量矩值。

```
set_Ixx (double val)/set_Iyy (double val)/set_Izz (double val)
```

- 获取负载在重心位置 XY 交叉方向的惯量积（单位：kg#mm²），返回值为 double 类型。

```
get_Ixy () const
```

- 设置负载在重心位置 XY 交叉方向的惯量积；val 为目标惯量积值。

```
set_Ixy (double val)
```

- 获取负载在重心位置 XZ 交叉方向的惯量积（单位：kg#mm²），返回值为 double 类型。

```
get_Ixz () const
```

- 设置负载在重心位置 XZ 交叉方向的惯量积；val 为目标惯量积值。

```
set_Ixz (double val)
```

- 获取负载在重心位置 YZ 交叉方向的惯量积（单位：kg#mm²），返回值为 double 类型。

```
get_Iyz () const
```

- 设置负载在重心位置 YZ 交叉方向的惯量积；val 为目标惯量积值。

```
set_Iyz (double val)
```

6.1.23 用户坐标系

功能介绍

E_ROB_COORD，用于定义机器人的用户坐标系参数，描述 TCP（工具中心点）相对于世界（基）坐标系的位移偏移和旋转欧拉角，提供参数的构造、赋值、数组批量操作及单个参数读写等完整接口。

参数说明

无

配置说明

- 默认构造函数，初始化后值无效。

```
E_ROB_COORD ()
```

- 基本构造函数；x 为 TCP 相对于世界（基）坐标系 X 方向位移偏移量（单位：mm），y 为 TCP 相对于世界（基）坐标系 Y 方向位移偏移量（单位：mm），z 为 TCP 相对于世界（基）坐标系 Z 方向位移偏移量（单位：mm），a 为 TCP 相对于世界（基）坐标系 z 轴旋转的欧拉角（单位：deg），b 为 TCP 相对于世界（基）坐标系 y’轴旋转的欧拉角（单位：deg），c 为 TCP 相对于世界（基）坐标系 x’’轴旋转的欧拉角（单位：deg）。

```
E_ROB_COORD (double x, double y, double z, double a, double b, double c)
```

- 数组构造函数；array 为坐标系值数组，格式为 {x,y,z,a,b,c}，debug 模式下不建议使用。

```
E_ROB_COORD(const std::vector<double>& array) / E_ROB_COORD (double array [6])
```

- 拷贝构造函数；t 为用户坐标系数据源对象。

```
E_ROB_COORD (const E_ROB_COORD& t)
```

成员函数

- 析构函数，释放类内资源。

```
~E_ROB_COORD ()
```

- 赋值运算符重载；t 为用户坐标系数据源对象，返回当前对象的引用。

```
E_ROB_COORD& operator=(const E_ROB_COORD& t)
```

- 批量设置用户坐标系参数；array 格式为 {x,y,z,a,b,c}，debug 模式下不建议使用。

```
setCoordByArray(const std::vector<double>& array) /setCoordByArray (double array [6])
```

- 获取 TCP 相对于世界（基）坐标系 X 方向的位移偏移量，返回值为 double 类型（单位：mm）。

```
getX () const
```

- 设置 TCP 相对于世界（基）坐标系 X 方向的位移偏移量；val 为目标偏移量值（单位：mm）。

```
setX (double val)
```

- 获取 TCP 相对于世界（基）坐标系 Y 方向的位移偏移量，返回值为 double 类型（单位：mm）。

```
getY () const
```

- 设置 TCP 相对于世界（基）坐标系 Y 方向的位移偏移量；val 为目标偏移量值（单位：mm）。

```
setY (double val)
```

- 获取 TCP 相对于世界（基）坐标系 Z 方向的位移偏移量，返回值为 double 类型（单位：mm）。

```
getZ () const
```

- 设置 TCP 相对于世界（基）坐标系 Z 方向的位移偏移量；val 为目标偏移量值（单位：mm）。

```
setZ (double val)
```

- 获取 TCP 相对于世界（基）坐标系 z 轴旋转的欧拉角，返回值为 double 类型（单位：deg）。

```
getA () const
```

- 设置 TCP 相对于世界（基）坐标系 z 轴旋转的欧拉角；val 为目标角度值（单位：deg）。

```
setA (double val)
```

- 获取 TCP 相对于世界（基）坐标系 y’轴旋转的欧拉角，返回值为 double 类型（单位：deg）。

```
getB () const
```

- 设置 TCP 相对于世界（基）坐标系 y’轴旋转的欧拉角；val 为目标角度值（单位：deg）。

```
setB (double val)
```

- 获取 TCP 相对于世界（基）坐标系 x’’轴旋转的欧拉角，返回值为 double 类型（单位：deg）。

```
getC () const
```

- 设置 TCP 相对于世界（基）坐标系 x’’轴旋转的欧拉角；val 为目标角度值（单位：deg）。

```
setC (double val)
```

6.2 接口列表

6.2.1 默认构造函数

功能介绍

用于初始化 EstunRemoteApiLib 类的实例对象，默认读取程序根目录下的 Config 文件夹。

参数说明

无

返回值

无

配置示例

```
EstunRemoteApiLib EstunApiObj();
```

6.2.2 有参构造函数

功能介绍

用于初始化 EstunRemoteApiLib 类的实例对象，支持手动指定配置文件。

参数说明

- configDir：配置文件及错误信息文件所在文件夹路径，数据类型为 ESString。

返回值

无

配置示例

```
EstunRemoteApiLib apiInstance("D:/yourPath/config");
```

6.2.3 有参构造函数

功能介绍

用于初始化 EstunRemoteApiLib 类的实例对象，支持手动指定配置文件和、具体的配置文件名，精准定位配置文件位置。

参数说明

- configDir：配置文件及错误信息文件所在文件夹路径，数据类型为 ESString；
- configFile：具体的配置文件名，数据类型为 ESString。

返回值

无

配置示例

创建 EstunRemoteApiLib 类的实例 apiInstance，指定配置文件所在文件夹为 "D:/your-Path"，具体配置文件名为 "config.xml"，后续该实例会从该路径下读取指定的配置文件完成初始化。

```
EstunRemoteApiLib apiInstance ("D:/yourPath", "config.xml");
```

6.2.4 设置本地日志配置

功能介绍

用于配置 EstunRemoteApiLib 类实例的本地日志存储规则，包括指定日志文件的存储目录和单个日志文件的大小限制。

参数说明

- logDir：日志文件存储目录路径，数据类型为 ESString，不指定则日志文件在程序运行当前目录生成。
- size：单个日志文件的大小限制，数据类型为 unsigned int。

返回值

无

配置示例

通过已创建的 EstunRemoteApiLib 实例 apiInstance 设置日志配置，指定日志文件存储在 "D:/RobotLogs/EstunApiLog" 目录下，单个日志文件大小限制为 1024000 字节（1MB），后续 API 运行产生的日志会按此规则存储。

```
apiInstance.setLogConfig ("D:/RobotLogs/EstunApiLog", 1024000);
```

6.2.5 日志打印开关

功能介绍

用于控制 EstunRemoteApiLib 类实例的本地日志输出开关，决定 writeLog 接口是否将日志内容写入文件。

参数说明

- enable：日志输出控制开关，数据类型为 bool；值为 true 时开启日志写入，值为 false 时关闭日志写入。

返回值

无

配置示例

通过已创建的 EstunRemoteApiLib 实例 apiInstance 开启本地日志写入功能，后续调用 writeLog 接口时，日志内容会被写入到指定的日志文件中；若传入 false 则关闭日志写入，writeLog 接口不会向文件写入任何内容。

```
apiInstance.setLogEnable (true);
```

6.2.6 写本地日志

功能介绍

用于将指定的日志信息写入本地日志文件。

参数说明

- str：待写入的日志信息内容，数据类型为 ESString。

返回值

无

配置示例

通过已创建的 EstunRemoteApiLib 实例 apiInstance 将 "连接机器人成功，IP：192.168.6.68" 这条日志信息写入本地日志文件；若 setLogEnable 设为 false，或未配置 logDir，日志内容不会实际写入文件。

```
apiInstance.writeLog ("连接机器人成功，IP：192.168.6.68");
```

6.2.7 错误日志

功能介绍

用于获取最近一次错误信息。

参数说明

无

返回值

返回最近一次错误信息字符串，数据类型为 ESString。

配置示例

通过已创建的 EstunRemoteApiLib 实例 apiInstance 获取最近一次错误信息，并将结果赋值给 ESString 类型变量 errMsg。

```
ESString errMsg = apiInstance.getLastErr ();
```

6.2.8 连接服务器

功能介绍

用于建立上位机与埃斯顿机器人之间的网络连接，支持配置自动重连机制，保障连接的稳定性。

参数说明

- ipAddr：机器人服务器的 IP 地址，需符合 IPV4 格式，数据类型为 ESString；
- isAutoReConnect：是否开启自动重连，数据类型为 bool，默认值为 false（异常掉线时不自动重连）。

返回值

连接成功返回 true，连接失败返回 false，数据类型为 bool。

配置示例

通过已创建的 EstunRemoteApiLib 实例 apiInstance，尝试连接 IP 地址为 "192.168.6.68" 的机器人服务器，并开启自动重连功能；若连接成功，变量 ret的值为 true，否则为 false；若省略第二个参数（如 apiInstance.connectToServer ("192.168.6.68")），则默认关闭自动重连。

```
bool iet = apiInstance.connectToServer("192.168.6.68", true);
```

6.2.9 断开连接

功能介绍

用于主动断开上位机与埃斯顿机器人之间已建立的网络连接，释放相关的网络资源。

参数说明

无

返回值

断开连接成功返回 true，断开失败返回 false，数据类型为 bool。

配置示例

通过已创建的 EstunRemoteApiLib 实例 apiInstance 主动断开与机器人服务器的连接；若断开操作成功，变量 isDisConnected 的值为 true，否则为 false；通常在程序退出、切换机器人连接目标等场景下调用该接口，确保网络资源正常释放。

```
bool isDisConnected = apiInstance.disConnect ();
```

6.2.10 连接检测

功能介绍

用于检测 EstunRemoteApiLib 类实例与埃斯顿机器人之间的网络连接状态，判断当前是否处于已连接状态。

参数说明

无

返回值

当前已连接机器人服务器返回 true，未连接 / 连接断开返回 false，数据类型为 bool。

配置示例

通过已创建的 EstunRemoteApiLib 实例 apiInstance 检测与机器人服务器的连接状态，结果赋值给 bool 类型变量 connectionState。

```
bool connectionState = apiInstance.isConnected ();
```

6.2.11 强制开启API模式接口

功能介绍

用于强制开启 / 关闭机器人控制器的 API 模式，该接口仅对拥有特殊授权的身份开放。

参数说明

- ident：特殊授权标识，数据类型为 ESString。
- enableStatus：API 模式控制状态，数据类型为 int，0 表示关闭 API 模式，1 表示开启 API 模式。

返回值

操作成功返回 true，失败返回 false，数据类型为 bool。

配置示例

通过已创建的 EstunRemoteApiLib 实例 apiInstance，使用特殊授权标识 "YourIdent" 强制开启机器人控制器的 API 模式；若操作成功，变量 isEnabled 的值为 true，否则为 false；调用该接口前需确保拥有对应的特殊授权。

```
bool isEnabled = apiInstance.E_ForceEnableApi("YourIdent", 1);
```

6.2.12 获取控制器时间接口

功能介绍

用于从埃斯顿机器人控制器中读取当前的系统时间，并以指定格式返回。

参数说明

- time_str：输出参数，用于接收控制器返回的时间字符串，数据类型为 ESString&，格式固定为 "YYYY-MM-DD hh:mm:ss"。

返回值

无

配置示例

6.2.13 获取当前模式的工作状态接口

功能介绍

用于查询机器人当前API模式下的工作状态，快速判断机器人是否正常运行、是否报错或处于停止状态。

参数说明

无

返回值

返回值为int类型：-1表示获取状态失败，0表示机器人工作正常，1表示机器人发生错误，2表示机器人处于停止状态（需调用start接口启动）

配置示例

通过apiInstance实例获取机器人API模式工作状态，根据返回值判断机器人状态；若返回2，需调用E_MotionStart接口唤醒机器人；若返回1，需先排查机器人错误再执行后续操作。

```
int apiStatus = apiInstance.E_GetAPIStatus();
```

6.2.14 申请机器人运动操作权限接口

功能介绍

用于向上位机系统申请机器人运动操作的权限，只有获取该权限后，才能执行机器人运动相关的控制指令。

参数说明

无

返回值

返回值为E_ROB_PERMIT类型，代表申请到的机器人运动操作权限信息。

配置示例

通过apiInstance实例申请机器人运动操作权限，返回的permit对象包含权限相关信息；后续执行运动控制指令前需确保成功获取该权限，权限使用完成后需调用E_ReleasePermit释放。

```
E_ROB_PERMIT permit = apiInstance.E_AcquirePermit();
```

6.2.15 查询当前机器人运动操作权限接口

功能介绍

用于查询当前系统中机器人运动操作权限的归属及状态，确认是否已获取权限、权限是否有效。

参数说明

无

返回值

返回值为E_ROB_PERMIT类型，代表当前机器人运动操作的权限信息。

配置示例

通过apiInstance实例查询当前机器人运动操作权限状态，可用于验证之前申请的权限是否仍有效，或判断是否有其他进程占用了运动权限。

```
E_ROB_PERMIT currentPermit = apiInstance.E_CurrentPermit();
```

6.2.16 释放机器人运动操作权限接口

功能介绍

用于释放之前通过E_AcquirePermit申请到的机器人运动操作权限。

参数说明

- permit：需释放的权限对象，数据类型为E_ROB_PERMIT（即之前申请到的权限信息）

返回值

释放权限成功返回true，失败返回false，数据类型为bool。

配置示例

先申请运动权限并执行相关操作，操作完成后调用该接口释放权限；若返回true表示权限释放成功，否则需检查permit是否为有效权限对象。

```
bool isReleased = apiInstance.E_ReleasePermit(permit);
```

6.2.17 唤醒机器人运动接口

功能介绍

用于将处于stop模式的机器人唤醒，使其从停止状态恢复为可运行状态。

参数说明

无

返回值

唤醒操作成功返回true，失败返回false，数据类型为bool。

配置示例

先判断机器人状态为停止，再调用E_MotionStart接口唤醒机器人；若返回true，机器人恢复可运行状态，可执行后续运动指令。

```
bool isStarted = apiInstance.E_MotionStart();
```

6.2.18 暂停机器人运动接口

功能介绍

用于暂停机器人当前的运动过程，暂停后不会清空已下发的运动队列，恢复后可继续执行剩余运动指令。

参数说明

无

返回值

暂停运动成功返回true，失败返回false，数据类型为bool。

配置示例

通过apiInstance实例暂停机器人运动，若返回true，机器人立即停止当前运动，但运动队列保留；适用于临时暂停运动（如人工干预、检测）的场景。

```
bool isPaused = apiInstance.E_MotionPause();
```

6.2.19 恢复机器人运动接口

功能介绍

用于恢复被E_MotionPause暂停的机器人运动，继续执行运动队列中未完成的运动指令。

参数说明

无

返回值

恢复运动成功返回true，失败返回false，数据类型为bool。

配置示例

机器人暂停后完成临时操作，调用该接口恢复运动；若返回true，机器人从暂停位置继续执行剩余的运动队列指令。

```
bool isContinued = apiInstance.E_MotionContinue();
```

6.2.20 停止机器人运动接口

功能介绍

用于强制停止机器人当前的所有运动，同时清空已下发的运动队列，需重新下发指令才能让机器人再次运动。

参数说明

无

返回值

停止运动并清空队列成功返回true，失败返回false，数据类型为bool。

配置示例

通过apiInstance实例停止机器人运动并清空运动队列，停止后需重新下发运动指令才能让机器人运行。

```
bool isStopped = apiInstance.E_MotionStop();
```

6.2.21 获取机器人运动状态接口

功能介绍

用于查询机器人当前的运动状态，判断机器人是否处于运动中。

参数说明

无

返回值

返回值为 int 类型：1 表示机器人正在运动，0 表示机器人未运动。

配置示例

通过 apiInstance 实例获取机器人运动状态，若返回 1 则说明机器人处于运动过程中，此时不建议下发冲突的运动指令；若返回 0 则可执行新的运动指令下发操作。

```
int runStatus = apiInstance.E_GetRobotRunStatus ();
```

6.2.22 运动下发（协议接口）接口

功能介绍

作为底层协议接口，用于下发关节运动或直线运动指令到机器人控制器，直接指定目标点位完成基础运动控制。

参数说明

- movType：运动类型标识，数据类型为 int，0 表示关节运动，1 表示直线运动；
- dstPoint ： 目标点位，数据类型为 E_ROB_POS&，支持关节角坐标或笛卡尔坐标格式

返回值

返回值为 int 类型：0 表示指令成功下发，-1 表示通信失败，-2 表示系统错误。

配置示例

下发直线运动（movType=1）指令到机器人，目标点位为 targetPos；根据返回值判断下发结果，返回 0 则指令下发成功，返回 - 1 需检查网络连接，返回 - 2 需排查机器人系统状态。

```
E_ROB_POS targetPos;
int ret = apiInstance.E_MovTcp (1, targetPos);
```

6.2.23 执行MovJ指令接口

功能介绍

用于执行机器人关节运动指令，通过指定点位、速度、过渡类型等变量名完成关节空间的运动控制。

参数说明

- Pname：目标位置点变量名，数据类型为 ESString，例："t_g.p1"。
- velName：运动速度变量名，数据类型为 ESString，默认空，例："t_g.SPEED0"。
- blendType：过渡类型，数据类型为 ESString，默认空，可选值为 ""、"Rel"、"Abs"。
- zname：过渡参数变量名，数据类型为 ESString，默认空，例："t_g.ZONE0"。
- tname：工具变量名，数据类型为 ESString，默认空，例："t_g.TOOL1"。
- coorname：坐标系变量名，数据类型为 ESString，默认空，例："t_g.USER0"。
- payloadName：负载变量名，数据类型为 ESString，默认空，例："t_g.PAYLOAD0"。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_-ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

执行关节运动指令，指定目标点位为 t_g.p1、速度为 t_g.SPEED0、相对过渡类型、过渡参数 t_g.ZONE0 等，到位精度 0.1，等待运动完成后接口返回；若 jobId 的 id≥0 则指令执行成功，否则失败。

```
E_ROB_JOBID jobId = apiInstance.E_MovJ ("t_g.p1", "t_g.SPEED0",
"Rel", "t_g.ZONE0", "t_g.TOOL1", "t_g.USER0", "t_g.PAYLOAD0", E_ROB_APIBUFF (), 0.1, true);
```

6.2.24 执行MovJ2指令接口

功能介绍

用于执行机器人关节运动指令，直接传入点位、速度、过渡参数等实体对象，而非变量名，更灵活地完成关节运动控制。

参数说明

- dstPoint：目标位置点，数据类型为 const E_ROB_POS&；velo 运行速度，数据类型为 const E_ROB_SPEED&，默认空。
- zone：过渡参数，数据类型为 const E_ROB_ZONE&，默认空。
- payloadId：负载号，数据类型为 int，默认 0；toolId 工具号，数据类型为 int，默认 0。
- coordId：用户坐标号，数据类型为 int，默认 0。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_-ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

直接传入目标点位、速度、过渡参数等实体对象执行关节运动，指定负载号 0、工具号 1、用户坐标号 0，到位精度 0.1，等待运动完成；jobId 的 id≥0 表示指令执行成功。

```
E_ROB_POS targetPos;
E_ROB_SPEED speed;
E_ROB_ZONE zone;
E_ROB_JOBID jobId = apiInstance.E_MovJ2 (targetPos, speed, zone, 0, 1, 0, E_ROB_APIBUFF (), 0.1, true);
```

6.2.25 执行MovL指令接口

功能介绍

用于执行机器人直线运动指令，通过指定点位、速度、过渡类型等变量名完成笛卡尔空间的直线运动控制。

参数说明

- Pname：目标位置点变量名，数据类型为 ESString，例："t_g.p1"。
- velName：运动速度变量名，数据类型为 ESString，默认空，例："t_g.SPEED0"。
- blendType：过渡类型，数据类型为 ESString，默认空，可选值为 ""、"Rel"、"Abs"。
- zname：过渡参数变量名，数据类型为 ESString，默认空，例："t_g.ZONE0"。
- tname：工具变量名，数据类型为 ESString，默认空，例："t_g.TOOL1"。
- coorname：坐标系变量名，数据类型为 ESString，默认空，例："t_g.USER0"。

- payloadName：负载变量名，数据类型为 ESString，默认空，例："t_g.PAYLOAD0"。
- overrideType：速率类型，数据类型为 ESString，默认"GON"，可选值为""、"GOFF"、"GON"。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

执行直线运动指令，指定目标点位 t_g.p2、绝对过渡类型、速率类型 GON，不等待运动完成立即返回；通过 jobId 可后续查询指令执行状态，id ≥ 0 表示指令下发成功。

```
E_ROB_JOBID jobId = apiInstance.E_MovL ("t_g.p2", "t_g.SPEED1",
 "Abs", "t_g.ZONE1", "t_g.TOOL2", "t_g.USER1", "t_g.PAYLOAD1", "GON", E_ROB_APIBUFF (), 0.05, false);
```

6.2.26 执行MovL2指令接口

功能介绍

用于执行机器人直线运动指令，直接传入点位、速度、过渡参数等实体对象，而非变量名，灵活完成笛卡尔空间直线运动控制。

参数说明

- dstPoint：目标位置点，数据类型为 const E_ROB_POS&。
- velo：运行速度，数据类型为 const E_ROB_SPEED&，默认空。
- zone：过渡参数，数据类型为 const E_ROB_ZONE&，默认空。
- payloadId：负载号，数据类型为 int，默认 0。
- toolId：工具号，数据类型为 int，默认 0。
- coordId：用户坐标号，数据类型为 int，默认 0。
- overrideType：速率类型，数据类型为 int，默认 1（0:GOVROFF，1:GOVRON）。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

执行直线运动指令，指定速率类型为 GOVRON（1）、用户坐标号 2，到位精度 0.05，等待运动完成；jobId 的 Id ≥ 0 表示指令执行成功。

```
E_ROB_POS targetPos;
E_ROB_SPEED speed;
E_ROB_JOBID jobId = apiInstance.E_MovL2 (targetPos, speed, E_ROB_ZONE (), 0, 1, 2, 1, E_ROB_APIBUFF
 (), 0.05, true);
```

6.2.27 执行MovC指令接口

功能介绍

用于执行机器人圆弧运动指令，通过指定辅助点、目标点等变量名完成圆弧轨迹的运动控制。

参数说明

- PAuxname：辅助位置点变量名，数据类型为 ESString，例："t_g.p0"。
- PDstname：目标位置点变量名，数据类型为 ESString，例："t_g.p1"。
- velName：运动速度变量名，数据类型为 ESString，默认空，例："t_g.SPEED0"。
- blendType：过渡类型，数据类型为 ESString，默认空，可选值为 ""、"Rel"、"Abs"。
- zname：过渡参数变量名，数据类型为 ESString，默认空，例："t_g.ZONE0"。
- tname：工具变量名，数据类型为 ESString，默认空，例："t_g.TOOL1"。
- coorname：坐标系变量名，数据类型为 ESString，默认空，例："t_g.USER0"。
- payloadName：负载变量名，数据类型为 ESString，默认空，例："t_g.PAYLOAD0"。
- overrideType：速率类型，数据类型为 ESString，默认"GON"，可选值为""、"GOF-F"、"GON"。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_-ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

执行圆弧运动指令，指定辅助点 t_g.p0、目标点 t_g.p3，相对过渡类型，等待运动完成；jobId 的 id≥0 表示圆弧运动指令执行成功。

```
E_ROB_JOBID jobId = apiInstance.E_MovC ("t_g.p0", "t_g.p3", "t_g.SPEED2",
"Rel", "t_g.ZONE2", "t_g.TOOL1", "t_g.USER0", "t_g.PAYLOAD0", "GON", E_ROB_APIBUFF (), 0.1, true);
```

6.2.28 执行MovC2指令接口

功能介绍

用于执行机器人圆弧运动指令，直接传入辅助点、目标点、速度等实体对象，而非变量名，灵活完成圆弧轨迹运动控制。

参数说明

- auxP：辅助位置点，数据类型为 const E_ROB_POS&。
- dstPoint：目标位置点，数据类型为 const E_ROB_POS&。
- velo：运行速度，数据类型为 const E_ROB_SPEED&，默认空。
- zone：过渡参数，数据类型为 const E_ROB_ZONE&，默认空。
- payloadId：负载号，数据类型为 int，默认 0。
- toolId：工具号，数据类型为 int，默认 0。
- coordId：用户坐标号，数据类型为 int，默认 0。
- overrideType：速率类型，数据类型为 int，默认 1（0:GOVROFF，1:GOVRON）。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_-ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

直接传入辅助点、目标点等实体对象执行圆弧运动，指定速率类型 GOVRON，等待运动完成；jobId 的 id≥0 表示指令执行成功。

```
E_ROB_POS auxPos, targetPos;
E_ROB_SPEED speed;
E_ROB_JOBID jobId = apiInstance.E_MovC2 (auxPos, targetPos, speed, E_ROB_ZONE (), 0, 1, 0, 1,
 E_ROB_APIBUFF (), 0.1, true);
```

6.2.29 执行MovArch指令接口

功能介绍

用于执行机器人拱形运动指令，通过指定点位、抬升高度等参数，让机器人以拱形轨迹运动到目标点，适用于避障场景。

参数说明

- Pname：目标位置点变量名，数据类型为 ESString，例："t_g.p1"。
- velName：运动速度变量名，数据类型为 ESString，默认空，例："t_g.SPEED0"。
- blendType：过渡类型，数据类型为 ESString，默认空，可选值为 ""、"Rel"、"Abs"。
- zname：过渡参数变量名，数据类型为 ESString，默认空，例："t_g.ZONE0"。
- tname：工具变量名，数据类型为 ESString，默认空，例："t_g.TOOL1"。
- coorname：坐标系变量名，数据类型为 ESString，默认空，例："t_g.USER0"。
- payloadName：负载变量名，数据类型为 ESString，默认空，例："t_g.PAYLOAD0"。
- cornerName：拱形运动转角变量名，数据类型为 ESString，默认空，例："t_g.ZONE1"。
- height：抬升高度，数据类型为 double，默认 0.0，单位毫米。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0；height2 第二抬升高度，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

执行拱形运动指令，指定目标点位 t_g.p4、抬升高度 50 毫米、第二抬升高度 30 毫米，绝对过渡类型，等待运动完成；jobId 的 id ≥ 0 表示拱形运动指令执行成功，机器人将以拱形轨迹避障运动到目标点。

```
E_ROB_JOBID jobId = apiInstance.E_MovArch ("t_g.p4", "t_g.SPEED3",
"Abs", "t_g.ZONE3", "t_g.TOOL2", "t_g.USER1", "t_g.PAYLOAD1", "t_g.ZONE1", 50.0, E_ROB_APIBUFF (),
 0.05, 30.0, true);
```

6.2.30 执行MovArch3指令接口

功能介绍

用于执行机器人拱形运动指令，直接传入目标点位、抬升高度、转角参数等实体对象，而非变量名，灵活完成拱形避障运动控制。

参数说明

- dstPoint：目标位置点，数据类型为 const E_ROB_POS&。
- velo：运行速度，数据类型为 const E_ROB_SPEED&，默认空。
- zone：过渡参数，数据类型为 const E_ROB_ZONE&，默认空。
- connerZone：转角参数，数据类型为 const E_ROB_ZONE&，默认空。
- payloadId：负载号，数据类型为 int，默认 0：
- toolId：工具号，数据类型为 int，默认 0。
- coordId：用户坐标号，数据类型为 int，默认 0。
- heigt：抬升高度，数据类型为 double，默认 0.0，单位毫米。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_-ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0。
- heigt2：第二抬升高度，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

直接传入实体对象执行拱形运动，指定抬升高度 40 毫米、第二抬升高度 20 毫米，等待运动完成；jobId 的 id≥0 表示指令执行成功，机器人按拱形轨迹运动到目标点。

```
E_ROB_POS targetPos;
E_ROB_SPEED speed;
E_ROB_ZONE zone, cornerZone;
E_ROB_JOBID jobId = apiInstance.E_MovArch3 (targetPos, speed, zone, cornerZone,
 0, 1, 0, 40.0, E_ROB_APIBUFF (), 0.05, 20.0, true);
```

6.2.31 执行MovArch2指令接口

功能介绍

用于执行机器人拱形运动指令，是 MovArch3 的简化版本，减少了第二抬升高度参数，直接传入实体对象完成拱形避障运动控制。

参数说明

- dstPoint：目标位置点，数据类型为 const E_ROB_POS&。
- velo：运行速度，数据类型为 const E_ROB_SPEED&，默认空。
- zone：过渡参数，数据类型为 const E_ROB_ZONE&，默认空。
- connerZone：转角参数，数据类型为 const E_ROB_ZONE&，默认空。
- payloadId：负载号，数据类型为 int，默认 0；toolId 工具号，数据类型为 int，默认 0。
- coordId：用户坐标号，数据类型为 int，默认 0。
- heigt：抬升高度，数据类型为 double，默认 0.0，单位毫米。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_-ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

执行简化版拱形运动指令，指定抬升高度 60 毫米，等待运动完成；jobId 的 id ≥ 0 表示指令执行成功，机器人以拱形轨迹运动到目标点位。

```
E_ROB_POS targetPos;
E_ROB_SPEED speed;
E_ROB_ZONE zone, cornerZone;
E_ROB_JOBID jobId = apiInstance.E_MovArch2 (targetPos, speed, zone, cornerZone, 0, 1, 0, 60.0,
 E_ROB_APIBUFF (), 0.1, true);
```

6.2.32 执行E_MovJArch2指令接口

功能介绍

用于执行机器人关节型拱形运动指令，通过指定抬升 / 下降高度、目标点位等参数，让机器人以关节运动的方式沿拱形轨迹运动到目标点，适用于避障场景。

参数说明

- dstPoint：目标位置点，数据类型为 const E_ROB_POS&。
- velo：运行速度，数据类型为 const E_ROB_SPEED&，默认空。
- zone：过渡参数，数据类型为 const E_ROB_ZONE&，默认空。
- connerZone：转角参数，数据类型为 const E_ROB_ZONE&，默认空。
- payloadId：负载号，数据类型为 int，默认 0。
- toolId：工具号，数据类型为 int，默认 0。
- coordId：用户坐标号，数据类型为 int，默认 0。
- Upheight：抬升高度，数据类型为 double，默认 0.0，单位毫米。
- Downheight：下降高度，数据类型为 double，默认 0.0，单位毫米。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_-ADDDO_IO 类型。
- valfine：到位精度值，数据类型为 double，默认 0.0。
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

执行关节型拱形运动指令，指定抬升高度 50 毫米、下降高度 30 毫米，等待运动完成；jobId 的 id ≥ 0 表示指令执行成功，机器人以关节运动方式沿拱形轨迹到达目标点。

```
E_ROB_POS targetPos;
E_ROB_SPEED speed;
E_ROB_ZONE zone, cornerZone;
E_ROB_JOBID jobId = apiInstance.E_MovJArch2 (targetPos, speed, zone,
cornerZone, 0, 1, 0, 50.0, 30.0, E_ROB_APIBUFF (), 0.1, true);
```

6.2.33 执行E_MovLArch2指令接口

功能介绍

用于执行机器人直线型拱形运动指令，通过指定抬升 / 下降高度、目标点位等参数，让机器人以直线运动的方式沿拱形轨迹运动到目标点，适用于避障场景。

参数说明

- dstPoint：目标位置点，数据类型为 const E_ROB_POS&。
- velo：运行速度，数据类型为 const E_ROB_SPEED&，默认空。

- zone：过渡参数，数据类型为 const E_ROB_ZONE&，默认空。
- connerZone：转角参数，数据类型为 const E_ROB_ZONE&，默认空。
- payloadId：负载号，数据类型为 int，默认 0；toolId 工具号，数据类型为 int，默认 0。
- coordId：用户坐标号，数据类型为 int，默认 0；Upheight 抬升高度，数据类型为 double，默认 0.0，单位毫米。
- Downheight：下降高度，数据类型为 double，默认 0.0，单位毫米。
- addDo：附加指令，数据类型为 const E_ROB_APIBUFF&，默认空，仅支持 E_ROB_-ADDDO_IO 类型：
- valfine：到位精度值，数据类型为 double，默认 0.0：
- IsWaitFinish：是否等待运动结束，数据类型为 bool，默认 false。

返回值

返回值为 E_ROB_JOBID 类型，代表指令标识信息，id 小于 0 表示指令执行失败。

配置示例

执行直线型拱形运动指令，指定抬升高度 60 毫米、下降高度 40 毫米，到位精度 0.05，等待运动完成；jobId 的 id≥ 0 表示指令执行成功，机器人以直线运动方式沿拱形轨迹到达目标点。

```
E_ROB_POS targetPos;
E_ROB_SPEED speed;
E_ROB_ZONE zone, cornerZone;
E_ROB_JOBID jobId = apiInstance.E_MovLArch2 (targetPos, speed, zone,
cornerZone, 0, 1, 0, 60.0, 40.0, E_ROB_APIBUFF (), 0.05, true);
```

6.2.34 获取所有工具坐标系ID接口

功能介绍

用于获取机器人控制器中已配置的所有工具坐标系的 ID，接口会自动初始化 ID 数组指针并返回 ID 数量。

参数说明

- num：输出参数，数据类型为 int&，用于接收获取到的工具坐标系 ID 个数。
- idArray：输出参数，数据类型为 int*&，ID 数组指针，由接口初始化。

返回值

成功获取所有工具坐标系 ID 返回 true，失败返回 false，数据类型为 bool。

配置示例

调用接口获取所有工具坐标系 ID，toolCount 接收 ID 个数，toolIds 接收 ID 数组指针；若 isSuccess 为 true，可遍历 toolIds 数组获取所有工具 ID（如 toolIds [0]、toolIds [1]）。

```
int toolCount = 0;int* toolIds = nullptr;
bool isSuccess = apiInstance.E_GetToolsID (toolCount, toolIds);
```

6.2.35 创建工具坐标系接口

功能介绍

用于在机器人控制器中创建新的工具坐标系，需指定唯一的工具号及对应的坐标、负载参数。

参数说明

- toolID 新的工具号，数据类型为 int，必须大于 0。
- toolCoord 工具坐标参数，数据类型为 const E_ROB_COORD&。
- payloadvalue 工具负载参数，数据类型为 const E_ROB_PAYLOAD&。

返回值

创建工具坐标系成功返回 true，失败返回 false，数据类型为 bool。

配置示例

创建工具号为 1 的工具坐标系，指定工具坐标和负载参数；若 isCreated 为 true，工具坐标系创建成功，可后续加载使用。

```
E_ROB_COORD toolCoord (10.0, 5.0, 8.0, 0.0, 0.0, 0.0);
E_ROB_PAYLOAD payload (2.5, 0.0, 0.0, 0.0);
bool isCreated = apiInstance.E_CreateTool (1, toolCoord, payload);
```

6.2.36 修改工具坐标系接口

功能介绍

用于修改机器人控制器中已存在的工具坐标系参数，需指定要修改的工具号及新的坐标、负载参数。

参数说明

- toolID：要修改的工具 id，数据类型为 int，必须大于 0。
- toolCoord：新的工具坐标参数，数据类型为 const E_ROB_COORD&。
- payloadvalue：新的工具负载参数，数据类型为 const E_ROB_PAYLOAD&。

返回值

修改工具坐标系成功返回 true，失败返回 false，数据类型为 bool。

配置示例

修改工具号为 1 的工具坐标系参数，替换为新的坐标和负载；若 isModified 为 true，修改操作成功。

```
E_ROB_COORD newToolCoord (12.0, 6.0, 9.0, 0.0, 0.0, 0.0);
E_ROB_PAYLOAD newPayload (3.0, 0.0, 0.0, 0.0);
bool isModified = apiInstance.E_ModifyTool (1, newToolCoord, newPayload);
```

6.2.37 加载工具坐标系接口

功能介绍

用于将指定 ID 的工具坐标系加载为机器人当前使用的工具坐标系，传入 0 则表示不加载任何工具坐标系。

参数说明

- toolID：工具 id，数据类型为 int，为 0 时表示不加载。

返回值

加载工具坐标系成功返回 true，失败返回 false，数据类型为 bool。

配置示例

加载工具号为 1 的工具坐标系为当前使用的工具坐标系；若 isLoaded 为 true，加载成功，后续机器人运动将使用该工具坐标系。

```
bool isLoaded = apiInstance.E_LoadTool (1);
```

6.2.38 获取当前工具ID接口

功能介绍

用于查询机器人当前正在使用的工具坐标系 ID。

参数说明

无

返回值

返回当前使用的工具 id，数据类型为 int。

配置示例

获取机器人当前加载的工具坐标系 ID，若返回 1 则表示当前使用工具号为 1 的工具坐标系。

```
int curToolId = apiInstance.E_GetCurToolID ();
```

6.2.39 获取工具坐标变量名称接口

功能介绍

用于查询指定工具 ID 对应的工具坐标变量名称。

参数说明

- toolID 工具 id，数据类型为 int。

返回值

返回工具坐标变量名称，数据类型为 ESString；若获取失败则返回空字符串。

配置示例

查询工具号为 1 的工具坐标变量名称，若返回 "t_g.TOOL1" 则表示该工具坐标系对应的变量名为 t_g.TOOL1，返回空则表示获取失败。

```
ESString toolName = apiInstance.E_GetToolName (1);
```

6.2.40 获取工具坐标系变量接口

功能介绍

用于查询指定工具 ID 对应的工具坐标系参数（坐标 + 负载），通过引用参数返回具体数值。

参数说明

- toolID：工具 id，数据类型为 int。
- toolCoord：输出参数，数据类型为 E_ROB_COORD&，接收工具坐标参数。
- toolPayload：输出参数，数据类型为 E_ROB_PAYLOAD&，接收工具负载参数。

返回值

获取工具坐标系变量成功返回 true，失败返回 false，数据类型为 bool。

配置示例

查询工具号为 1 的工具坐标系参数，若 isGot 为 true，toolCoord 和 toolPayload 会被赋值为该工具的坐标、负载参数；失败则参数无有效内容。

```
E_ROB_COORD toolCoord;E_ROB_PAYLOAD toolPayload;
bool isGot = apiInstance.E_GetTool (1, toolCoord, toolPayload);
```

6.2.41 获取所有用户坐标系ID接口

功能介绍

用于获取机器人控制器中已配置的所有用户坐标系 ID、，接口会自动初始化 ID 数组指针并返回 ID 数量。

参数说明

- num：输出参数，数据类型为 int&，用于接收获取到的用户坐标系 ID 个数。
- idArray：输出参数，数据类型为 int*&，ID 数组指针，由接口初始化。

返回值

成功获取所有用户坐标系 ID 返回 true，失败返回 false，数据类型为 bool。

配置示例

调用接口获取所有用户坐标系 ID，coordCount 接收 ID 个数，coordIds 接收 ID 数组指针；若 isSuccess 为 true，可遍历 coordIds 数组获取所有用户 ID。

```
int coordCount = 0;
int* coordIds = nullptr;
bool isSuccess = apiInstance.E_GetUserCoordsID (coordCount, coordIds);
```

6.2.42 创建用户坐标接口

功能介绍

用于在机器人控制器中创建新的用户坐标系，需指定唯一的坐标 ID及坐标参数。

参数说明

- usrID：新的坐标 id，数据类型为 int，必须大于 0。
- usrData：坐标参数，数据类型为 const E_ROB_COORD&。

返回值

创建用户坐标成功返回 true，失败返回 false，数据类型为 bool。

配置示例

创建用户坐标号为 1 的用户坐标系；若要创建外部坐标系，可传入真实 ID，usrData 为外部坐标系参数。

```
E_ROB_COORD userCoord (50.0, 30.0, 20.0, 0.0, 0.0, 0.0);
bool isCreated = apiInstance.E_CreateUserCoord (1, userCoord);
```

6.2.43 修改用户坐标接口

功能介绍

用于修改机器人控制器中已存在的用户坐标系参数，需指定要修改的坐标 ID及新的坐标参数。

参数说明

- usrID。要修改的用户坐标 id，数据类型为 int，必须大于 0。
- usrData。新的坐标参数，数据类型为 const E_ROB_COORD&。

返回值

修改用户坐标成功返回 true，失败返回 false，数据类型为 bool。

配置示例

修改用户坐标号为 1 的用户坐标系参数；若修改外部坐标系，传入 10001 即可。

```
E_ROB_COORD newUserCoord (55.0, 35.0, 25.0, 0.0, 0.0, 0.0);
bool isModified = apiInstance.E_ModifyUserCoord (1, newUserCoord);
```

6.2.44 加载用户坐标接口

功能介绍

用于将指定 ID 的用户坐标系加载为机器人当前使用的用户坐标系，传入 0 则表示不加载任何用户坐标系。

参数说明

- usrID：用户坐标 id，数据类型为 int，为 0 时表示不加载。

返回值

加载用户坐标成功返回 true，失败返回 false，数据类型为 bool。

配置示例

加载用户坐标号为 1 的用户坐标系为当前使用的坐标系。

```
bool isLoaded = apiInstance.E_LoadUserCoord (1);
```

6.2.45 获取当前用户坐标ID接口

功能介绍

用于查询机器人当前正在使用的用户坐标系 ID（外部坐标系的 ID 为真实 ID+10000），同时返回用户坐标类型。

参数说明

- coordtype：输出参数，数据类型为 E_UserCoordType&，接收用户坐标类型。

返回值

返回当前使用的用户坐标号，数据类型为 int。

配置示例

获取当前用户坐标系 ID 和类型，若 curCoordId 为 1、coordType 为本地坐标系类型，则表示当前使用本地用户坐标 1；若 curCoordId 为 10001，则表示使用外部坐标系真实 ID1。

```
E_UserCoordType coordType;
int curCoordId = apiInstance.E_GetCurUserCoordID (coordType);
```

6.2.46 获取用户坐标变量名称接口

功能介绍

用于查询指定用户坐标 ID 和类型对应的用户坐标变量名称（外部坐标系的 ID 为真实 ID +10000）。

参数说明

- coordID：用户坐标 id，数据类型为 int（外部坐标系的 ID 为真实 ID+10000）。
- coordtype：用户坐标类型，数据类型为 E_UserCoordType。

返回值

返回用户坐标变量名称，数据类型为 ESString；若获取失败则返回空字符串。

配置示例

查询用户坐标号为 1、指定类型的用户坐标变量名称，若返回 "t_g.USER0" 则表示该坐标系对应的变量名，返回空则获取失败。

```
E_UserCoordType coordType;ESString coordName =
apiInstance.E_GetUserCoordName (1, coordType);
```

6.2.47 获取用户坐标信息接口

功能介绍

用于查询指定用户坐标 ID 和类型对应的用户坐标系参数，通过引用参数返回具体数值（外部坐标系的 ID 为真实 ID+10000）。

参数说明

- coordID：用户坐标 id，数据类型为 int（外部坐标系的 ID 为真实 ID+10000）。
- coordtype：用户坐标类型，数据类型为 E_UserCoordType。
- coord：输出参数，数据类型为 E_ROB_COORD&，接收用户坐标参数。

返回值

获取用户坐标信息成功返回 true，失败返回 false，数据类型为 bool。

配置示例

查询用户坐标号为 1、指定类型的用户坐标系参数，若 isGot 为 true，userCoord 会被赋值为该坐标系的具体参数；失败则参数无有效内容。

```
E  UserCoordType coordType;E  ROB  COORD userCoord;
bool isGot = apiInstance.E_GetUserCoord (1, coordType, userCoord);
```

6.2.48 获取所有负载ID接口

功能介绍

用于获取机器人控制器中已配置的所有负载的 ID，接口会自动初始化 ID 数组指针并返回 ID 数量。

参数说明

- num：输出参数，数据类型为 int&，用于接收获取到的负载 ID 个数。
- idArray：输出参数，数据类型为 int*&，ID 数组指针，由接口初始化。

返回值

成功获取所有负载 ID 返回 true，失败返回 false，数据类型为 bool。

配置示例

调用接口获取所有负载 ID，payloadCount 接收 ID 个数，payloadIds 接收 ID 数组指针；若 isSuccess 为 true，可遍历 payloadIds 数组获取所有负载 ID，使用后需释放 payloadIds 内存。

```
int payloadCount = 0;int* payloadIds = nullptr;
bool isSuccess = apiInstance.E_GetPayLoadsID (payloadCount, payloadIds);
```

6.2.49 创建负载接口

功能介绍

用于在机器人控制器中创建新的负载参数，需指定唯一的负载 ID 及对应的负载参数。

参数说明

- payloadID：新的负载 id，数据类型为 int，必须大于 0。
- payloadData：负载参数，数据类型为 const E_ROB_PAYLOAD&。

返回值

创建负载成功返回 true，失败返回 false，数据类型为 bool。

配置示例

创建负载号为 1 的负载参数；若 isCreated 为 true，负载创建成功，可后续加载使用。

```
E_ROB_PAYLOAD payload (5.0, 0.0, 0.0, 10.0);
bool isCreated = apiInstance.E_CreatePayLoad (1, payload);
```

6.2.50 修改负载接口

功能介绍

用于修改机器人控制器中已存在的负载参数，需指定要修改的负载 ID 及新的负载参数。

参数说明

- payloadID：要修改的负载 id，数据类型为 int，必须大于 0。
- payloadData：新的负载参数，数据类型为 const E_ROB_PAYLOAD&。

返回值

修改负载成功返回 true，失败返回 false，数据类型为 bool。

配置示例

修改负载号为 1 的负载参数，替换为新的参数；若 isModified 为 true，修改操作成功。

```
E_ROB_PAYLOAD newPayload (6.0, 0.0, 0.0, 12.0);
bool isModified = apiInstance.E_ModifyPayLoad (1, newPayload);
```

6.2.51 加载负载接口

功能介绍

用于将指定 ID 的负载参数加载为机器人当前使用的负载，传入 0 则表示不加载任何负载。

参数说明

- payloadID：负载 id，数据类型为 int，为 0 时表示不加载。

返回值

加载负载成功返回 true，失败返回 false，数据类型为 bool。

配置示例

加载负载号为 1 的负载参数为当前使用的负载；若 isLoaded 为 true，加载成功，后续机器人运动将基于该负载参数计算。

```
bool isLoaded = apiInstance.E_LoadPayLoad (1);
```

6.2.52 获取当前负载变量ID接口

功能介绍

用于查询机器人当前正在使用的负载参数 ID。

参数说明

无

返回值

返回当前使用的负载号，数据类型为 int。

配置示例

获取机器人当前加载的负载 ID，若返回 1 则表示当前使用负载号为 1 的负载参数。

```
int curPayloadId = apiInstance.E_GetCurPayLoadID ();
```

6.2.53 获取负载变量名称接口

功能介绍

用于查询指定负载 ID 对应的负载变量名称。

参数说明

- payloadID：负载变量 id，数据类型为 int。

返回值

返回负载变量名称，数据类型为 ESString；若获取失败则返回空字符串。

配置示例

查询负载号为 1 的负载变量名称，若返回 "t_g.PAYLOAD0" 则表示该负载对应的变量名，返回空则获取失败。

```
ESString payloadName = apiInstance.E_GetPayLoadName (1);
```

6.2.54 获取负载变量接口

功能介绍

用于查询指定负载 ID 对应的负载参数，通过引用参数返回具体数值

参数说明

- payloadID：负载变量 id，数据类型为 int。
- payload：输出参数，数据类型为 E_ROB_PAYLOAD&，接收负载参数。

返回值

获取负载变量成功返回 true，失败返回 false，数据类型为 bool。

配置示例

查询负载号为 1 的负载参数，若 isGot 为 true，payload 会被赋值为该负载的重量、偏移量等参数；失败则参数无有效内容。

```
E_ROB_PAYLOAD payload;
bool isGot = apiInstance.E_GetPayLoad (1, payload);
```

6.2.55 获取默认加载的负载参数接口

功能介绍

用于查询机器人控制器中默认加载的负载参数，包括负载 ID、名称和具体参数，通过引用参数返回。

参数说明

- payloadID：输出参数，数据类型为 int&，接收默认负载 ID。
- name：输出参数，数据类型为 ESString&，接收默认负载名称。
- payload：输出参数，数据类型为 E_ROB_PAYLOAD&，接收默认负载参数。

返回值

获取默认加载的负载参数成功返回 true，失败返回 false，数据类型为 bool。

配置示例

查询机器人默认加载的负载参数，若 isGot 为 true，defPayloadId、defPayloadName、defPayload 会分别被赋值为默认负载的 ID、名称和具体参数；失败则参数无有效内容。

```
int defPayloadId;
ESString defPayloadName;
E_ROB_PAYLOAD defPayload;
bool isGot = apiInstance.E_GetDefaultPayload (defPayloadId, defPayloadName, defPayload);
```

6.2.56 设置IO的值接口

功能介绍

用于批量设置机器人指定类型、指定起始下标 IO 的数值，可一次性修改多个连续 IO 的状态。

参数说明

- IoIndex：开始的 IO 下标，数据类型为 int，从 1 开始。
- IoType IO：类型，数据类型为 E_IOType。
- dataIn：要修改的 IO 值数组指针，数据类型为 double*。
- dataNuml：要修改的 IO 值数组元素个数，数据类型为 int。

返回值

设置 IO 值成功返回 true，失败返回 false，数据类型为 bool。

配置示例

从下标 1 开始，设置 3 个数字输出型 IO 的值依次为 1、0、1；若 isSuccess 为 true，IO 值设置成功。

```
double ioValues [] = {1.0, 0.0, 1.0};
bool isSuccess = apiInstance.E_SetIOValue (1, E_IOType::DIGITAL_OUTPUT, ioValues, 3);
```

6.2.57 设置IO的值（带等待）接口

功能介绍

用于批量设置机器人指定类型、指定起始下标 IO 的数值，支持配置是否等待运动完成后再返回接口结果。

参数说明

- IoIndex：开始的 IO 下标，数据类型为 int，从 1 开始。
- IoType IO：类型，数据类型为 E_IOType。
- dataIn：要修改的 IO 值数组指针，数据类型为 double*。
- dataNum：要修改的 IO 值数组元素个数，数据类型为 int。
- finished：是否等待运动完成，数据类型为 bool，默认 true（等待）。

返回值

设置 IO 值成功返回 true，失败返回 false，数据类型为 bool。

配置示例

从下标 5 开始，设置 2 个数字输入型 IO 的值为 0、1，不等待运动完成立即返回；isSuccess 为 true 表示设置操作成功。

```
double ioValues [] = {0.0, 1.0};
bool isSuccess = apiInstance.E_SetIOValue_DC (5, E_IOType::DIGITAL_INPUT, ioValues, 2, false);
```

6.2.58 获取IO的值接口

功能介绍

用于批量获取机器人指定类型、指定起始下标、指定数量 IO 的数值，需预先为返回数组申请内存。

参数说明

- IoIndex：开始的 IO 下标，数据类型为 int，从 1 开始。
- IoType IO：类型，数据类型为 E_IOType。
- IoNum：查询 IO 数量，数据类型为 int（需和数组元素个数一致）。
- dataOut IO：值返回数组指针，数据类型为 double*，需预先申请内存。

返回值

获取 IO 值成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取从下标 1 开始的 3 个数字输出型 IO 的值，结果存储在 ioValues 数组中。

```
double* ioValues = new double [3];
bool isSuccess = apiInstance.E_GetIOValue (1, E_IOType::DIGITAL_OUTPUT, 3, ioValues);
```

6.2.59 设置IO备注接口

功能介绍

用于为机器人指定类型、指定下标的 IO 设置备注信息，便于标识 IO 用途。

参数说明

- IoType IO：类型，数据类型为 E_IOType。
- IoIndex IO：下标，数据类型为 int，从 1 开始。
- label：备注内容，数据类型为 ESString。

返回值

设置 IO 备注成功返回 true，失败返回 false，数据类型为 bool。

配置示例

为下标 2 的模拟输出型 IO 设置备注 “电压输出”；isSuccess 为 true 表示备注设置成功。

```
bool isSuccess = apiInstance.E_SetIONote (E_IOType::ANALOG_OUTPUT, 2, "电压输出");
```

6.2.60 获取IO备注接口

功能介绍

用于获取机器人指定类型 IO 的备注信息，支持获取单个 IO 备注或该类型所有 IO 备注（多个备注以 ';' 分隔），可指定语言类型。

参数说明

IoType IO 类型，数据类型为 E_IOType；

IoIndex IO 下标，数据类型为 int，从 1 开始，为 - 1 时获取该类所有 IO 备注；

label 备注内容返回值，数据类型为 ESString&（utf 格式，多个以 ';' 隔开）；

languageType 语言类型，数据类型为 const int，默认 0（中文），1 为英文

返回值

获取 IO 备注成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取所有数字输入型 IO 的英文备注，结果存储在 note 中（如 "StartBtn;StopBtn;EmergencyBtn"）；isSuccess 为 true 表示获取成功。

```
ESString note;
bool isSuccess = apiInstance.E_GetIONote (E_IOType::DIGITAL_INPUT, -1, note, 1);
```

6.2.61 刷新全局变量接口

功能介绍

用于从配置文件中刷新机器人的全局变量，确保内存中的全局变量值与文件中的最新值一致。

参数说明

无

返回值

刷新全局变量成功返回 true，失败返回 false，数据类型为 bool。

配置示例

调用接口刷新全局变量，isSuccess 为 true 表示刷新成功，全局变量会同步为文件中的最新值。

```
bool isSuccess = apiInstance.E_RefreshGlobalVarFromFile ();
```

6.2.62 设置变量是否掉电保存接口

功能介绍

用于配置机器人指定类型、指定名称、指定作用域的变量是否开启掉电保存功能，开启后变量值在断电重启后不会丢失。

参数说明

- varType：变量类型，数据类型为 E_VarType。
- varName：变量名，数据类型为 ESString。
- varScope：作用域，数据类型为 E_ScopeType。
- isSave：是否掉电保存，数据类型为 bool。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

设置全局作用域下的整型变量 g_speed 为掉电保存；isSuccess 为 true 表示配置生效。

```
bool isSuccess = apiInstance.E_SetVarIsSaveFile (E_VarType::INT, "g_speed", E_ScopeType::GLOBAL,
 true);
```

6.2.63 设置变量接口

功能介绍

用于修改机器人指定类型、指定名称、指定作用域的变量值，变量值需先转换为字符串类型传入。

参数说明

- varType：变量类型，数据类型为 E_VarType。
- varName：变量名，数据类型为 ESString。

- varValue：变量值，数据类型为 ESString（需从其他类型转换）。
- varScope：变量域，数据类型为 E_ScopeType。

返回值

设置变量值成功返回 true，失败返回 false，数据类型为 bool。

配置示例

设置局部作用域下的浮点型变量 g_height 的值为 50.5（以字符串 "50.5" 传入）；isSuccess 为 true 表示变量值修改成功。

```
bool isSuccess = apiInstance.E_SetVarValue (E_VarType::DOUBLE, "g_height", "50.5",
 E_ScopeType::LOCAL);
```

6.2.64 获取变量接口

功能介绍

用于获取机器人指定类型、指定名称、指定作用域的变量值，返回值为字符串类型，需自行转换为对应类型。

参数说明

- varType：变量类型，数据类型为 E_VarType。
- varName：变量名，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- valueBack：变量值返回值，数据类型为 ESString&（字符串类型，需转换为其他类型）。

返回值

获取变量值成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取全局作用域下的布尔型变量 g_isRun 的值，结果存储在 value 中（如 "true" 或 "false"）；isSuccess 为 true 表示获取成功。

```
ESString value;
bool isSuccess = apiInstance.E_GetVarValue (E_VarType::BOOL, "g_isRun", E_ScopeType::GLOBAL,
 value);
```

6.2.65 设置默认位置到精度值接口

功能介绍

用于设置机器人运动的默认到位精度值，该配置支持掉电保存，重启后仍生效。

参数说明

- value：到位精度值，数据类型为 double。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将机器人默认到位精度设置为 0.05mm；isSuccess 为 true 表示配置成功，后续运动指令未指定精度时会使用该默认值。

```
bool isSuccess = apiInstance.E_SetFine (0.05);
```

6.2.66 获取默认位置到精度值接口

功能介绍

用于获取机器人当前配置的默认到位精度值，通过引用参数返回具体数值。

参数说明

- value：到位精度值返回值，数据类型为 double&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取机器人默认到位精度值并存储在 fineValue 中；isSuccess 为 true 时，fineValue 为有效精度值（如 0.05）。

```
double fineValue;bool isSuccess = apiInstance.E_GetFine (fineValue);
```

6.2.67 设置点动坐标系类型接口

功能介绍

用于配置机器人点动操作时使用的坐标系类型（如关节坐标系、世界坐标系等）。

参数说明

- coordType：坐标系类型，数据类型为 E_CoordType。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将点动坐标系设置为关节坐标系；isSuccess 为 true 表示配置生效，后续点动操作基于关节坐标系执行。

```
bool isSuccess = apiInstance.E_SetJogCoord (E_CoordType::JOINT);
```

6.2.68 获取点动坐标系类型接口

功能介绍

用于查询机器人当前点动操作使用的坐标系类型。

参数说明

无

返回值

返回当前点动坐标系类型，数据类型为 E_CoordType。

配置示例

获取当前点动坐标系类型，若返回 E_CoordType::WORLD 则表示使用世界坐标系。

```
E_CoordType curCoord = apiInstance.E_GetJogCoord ();
```

6.2.69 设置jog是否可用接口

功能介绍

用于启用或禁用机器人的点动（jog）功能，禁用后无法执行点动操作。

参数说明

- isEnable：点动功能开关，数据类型为 bool，true 表示可用，false 表示不可用。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

启用机器人点动功能；isSuccess 为 true 表示配置成功，可执行后续点动指令。

```
bool isSuccess = apiInstance.E_SetJogEnable (true);
```

6.2.70 获取jog是否可用接口

功能介绍

用于查询机器人当前点动（jog）功能是否处于可用状态。

参数说明

无

返回值

点动功能可用返回 true，不可用返回 false，数据类型为 bool。

配置示例

获取点动功能状态，isJogEnable 为 true 时可执行点动操作，false 则不可。

```
bool isJogEnable = apiInstance.E_GetJogEnable ();
```

6.2.71 点动接口

功能介绍

用于控制机器人单轴点动运动，可指定运动轴、方向及启停状态。

参数说明

- axisId：轴编号，数据类型为 int，值 [0,5]（关节坐标系对应轴 1-6，世界坐标系对应 xyzabc）。
- distention：运动方向，数据类型为 E_DistentionType。
- mov：运动状态，数据类型为 bool，true 表示运动，false 表示停止。

返回值

操作成功返回 true，失败返回 false，数据类型为 bool。

配置示例

控制关节坐标系下轴 1（axisId=0）沿正方向点动；传入 false 可停止该轴点动。

```
bool isSuccess = apiInstance.E_JogMotion (0, E_DistentionType::POSITIVE, true);
```

6.2.72 寸动接口

功能介绍

用于控制机器人单轴以指定距离进行寸动运动，支持指定运动轴、方向、距离及启停状态。

参数说明

- axisId：轴编号，数据类型为 int，值 [0,5]。
- distention：运动方向，数据类型为 E_DistentionType。
- dist：寸动距离，数据类型为 double（单位毫米）。
- mov：运动状态，数据类型为 bool，true 表示运动，false 表示停止。

返回值

操作成功返回 true，失败返回 false，数据类型为 bool。

配置示例

控制关节坐标系下轴 2（axisId=1）沿负方向寸动 5 毫米；isSuccess 为 true 表示寸动指令执行成功。

```
bool isSuccess = apiInstance.E_JogMicroMotion (1, E_DistentionType::NEGATIVE, 5.0, true);
```

6.2.73 示教点接口

功能介绍

用于将机器人当前位置示教为指定名称、指定作用域的位置点变量，作用域不能为系统域。

参数说明

- pointName：位置点变量名，数据类型为 ESString。
- scope：变量域，数据类型为 E_ScopeType（不能是系统域）。

返回值

示教成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将机器人当前位置示教为局部作用域的位置点变量 p_teach_01；isSuccess 为 true 表示示教成功。

```
bool isSuccess = apiInstance.E_TeachPoint ("p_teach_01", E_ScopeType::LOCAL);
```

6.2.74 获取机器人类型接口

功能介绍

用于查询机器人的型号信息，通过引用参数返回具体型号字符串。

参数说明

- value：机器人型号返回值，数据类型为 ESString&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取机器人型号并存储在 robotType 中（如 "ER12-C10"）；isSuccess 为 true 表示获取成功。

```
ESString robotType;
bool isSuccess = apiInstance.E_GetRobotType (robotType);
```

6.2.75 获取机器人信息接口

功能介绍

用于查询机器人的详细信息（如序列号、硬件版本等），通过引用参数返回信息字符串。

参数说明

- value：机器人信息返回值，数据类型为 ESString&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取机器人详细信息并存储在 robotInfo 中；isSuccess 为 true 表示获取成功。

```
ESString robotInfo;bool isSuccess = apiInstance.E_GetRobotInfo (robotInfo);
```

6.2.76 获取本体轴数接口

功能介绍

用于查询机器人本体的轴数（如 6 轴机器人返回 6）。

参数说明

无

返回值

返回本体轴数，数据类型为 int；小于 0 表示获取失败。

配置示例

获取机器人本体轴数，axisNum 为 6 表示 6 轴机器人，为 - 1 表示获取失败。

```
int axisNum = apiInstance.E_GetRobotAxisNum ();
```

6.2.77 获取附加轴数接口

功能介绍

用于查询机器人配置的附加轴数量（如变位机轴、导轨轴等）。

参数说明

无

返回值

返回附加轴数，数据类型为 int；小于 0 表示获取失败。

配置示例

获取附加轴数，auxAxisNum 为 2 表示配置了 2 个附加轴，为 - 1 表示获取失败。

```
int auxAxisNum = apiInstance.E_GetAuxAxisNum ();
```

6.2.78 获取机器人状态接口

功能介绍

用于判断机器人是否在线，返回特定数值标识在线状态。

参数说明

无

返回值

返回值为 int 类型，12345 表示机器人在线，其余值表示不在线。

配置示例

判断机器人状态，robotStatus 为 12345 时表示在线，否则为离线。

```
int robotStatus = apiInstance.E_GetRobotStatus ();
```

6.2.79 设置全局速度接口

功能介绍

用于设置机器人运动的全局速度倍率，取值范围为 0-100（百分比）。

参数说明

- value：速度值，数据类型为 int，范围 [0-100]。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将机器人全局速度设置为 50%；isSuccess 为 true 表示配置生效，后续运动速度为指令速度的 50%。

```
bool isSuccess = apiInstance.E_SetGlobalSpeed (50);
```

6.2.80 获取全局速度接口

功能介绍

用于查询机器人当前的全局速度倍率值，返回 0 时需结合 getLastErr 排查是否出错。

参数说明

无

返回值

返回全局速度值，数据类型为 int；返回 0 可能是速度为 0 或获取失败，需调用 getLastErr 获取错误信息。

配置示例

获取全局速度倍率，globalSpeed 为 50 表示当前速度倍率是 50%。

```
int globalSpeed = apiInstance.E_GetGlobalSpeed ();
if (globalSpeed == 0)
{ESString err = apiInstance.getLastErr ();
```

6.2.81 设置系统模式接口

功能介绍

用于配置机器人的系统模式，目前仅支持手动模式和 API 模式，其余模式设置无效。

参数说明

- sysMode：系统模式，数据类型为 E_SysModeType。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将系统模式设置为 API 模式；isSuccess 为 true 表示配置生效，可通过 API 控制机器人。
补充：E_SetSysMode_ALL 接口功能与 E_SetSysMode 一致，仅命名不同，使用方式相同（如 apiInstance.E_SetSysMode_ALL (E_SysModeType::MANUAL)）。

```
bool isSuccess = apiInstance.E_SetSysMode (E_SysModeType::API);
```

6.2.82 获取系统模式接口

功能介绍

用于查询机器人当前的系统模式（手动 / API 等）。

参数说明

无

返回值

返回当前系统模式，数据类型为 E_SysModeType。

配置示例

获取系统模式，curMode 为 E_SysModeType::API 表示当前是 API 模式。

```
E_SysModeType curMode = apiInstance.E_GetSysMode ();
```

6.2.83 获取急停按钮状态接口

功能介绍

用于查询机器人急停按钮是否被按下。

参数说明

无

返回值

急停按钮按下返回 true，未按下返回 false，数据类型为 bool。

配置示例

判断急停状态，isEmergency 为 true 时表示急停按钮已按下，需解除急停后才能操作机器人。

```
bool isEmergency = apiInstance.E_GetEmergencyBtnStatus ();
```

6.2.84 设置运动模式接口

功能介绍

用于配置机器人的运动模式（如自动、手动、示教等）。

参数说明

- mode：运动模式，数据类型为 E_RunModeType。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将机器人运动模式设置为自动模式；isSuccess 为 true 表示配置生效。

```
bool isSuccess = apiInstance.E_SetRunMode (E_RunModeType::AUTO);
```

6.2.85 获取运动模式接口

功能介绍

用于查询机器人当前的运动模式。

参数说明

无

返回值

返回当前运动模式，数据类型为 E_RunModeType。

配置示例

获取运动模式，curRunMode 为 E_RunModeType::TEACH 表示当前是示教模式。

```
E_RunModeType curRunMode = apiInstance.E_GetCurRunMode ();
```

6.2.86 设置机器人伺服使能接口

功能介绍

用于控制机器人伺服系统的使能状态，上使能后机器人可运动，掉使能后无法运动。

参数说明

- isOn：使能状态，数据类型为 bool，true 表示上使能，false 表示掉使能

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

为机器人伺服系统上使能；isSuccess 为 true 表示使能成功，机器人可执行运动指令。

```
bool isSuccess = apiInstance.E_SetServoOn (true);
```

6.2.87 获取机器人伺服是否使能接口

功能介绍

用于查询机器人伺服系统的当前使能状态。

参数说明

无

返回值

返回伺服使能状态，数据类型为 E_ServoStatusType。

配置示例

获取伺服使能状态，servoStatus 为 E_ServoStatusType::ON 表示伺服已使能。

```
E_ServoStatusType servoStatus = apiInstance.E_GetServoOn ();
```

6.2.88 获取机器人待机励磁状态接口

功能介绍

用于查询机器人伺服系统的待机励磁状态。

参数说明

无

返回值

返回待机励磁状态，数据类型为 E_ServoStatusType。

配置示例

获取待机励磁状态，standbyStatus 为 E_ServoStatusType::ON 表示处于待机励磁状态。

```
E_ServoStatusType standbyStatus = apiInstance.E_GetStandbyMotState ();
```

6.2.89 设置震动抑制使能接口

功能介绍

用于启用或禁用机器人的震动抑制功能，启用后可降低运动过程中的震动。

参数说明

- enable：使能开关，数据类型为 bool，true 表示使能，false 表示不使能。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

启用震动抑制功能；isSuccess 为 true 表示配置生效，机器人运动时震动会被抑制。

```
bool isSuccess = apiInstance.E_SetMotionVibraEnable (true);
```

6.2.90 获取震动抑制使能状态接口

功能介绍

用于查询机器人震动抑制功能的当前使能状态，通过引用参数返回结果。

参数说明

- enable：使能状态返回值，数据类型为 bool&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取震动抑制使能状态，vibraEnable 为 true 表示功能已启用。

```
bool vibraEnable;bool isSuccess = apiInstance.E_GetMotionVibraEnable (vibraEnable);
```

6.2.91 设置震动抑制等级接口

功能介绍

用于为机器人指定轴（1-6）配置震动抑制等级，等级不同抑制效果不同。

参数说明

- axis：轴号，数据类型为 int，范围 1-6。
- enumValue：震动抑制等级，数据类型为 E_SuppresLevel。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将机器人 3 轴的震动抑制等级设置为 2 级；isSuccess 为 true 表示配置生效。

```
bool isSuccess = apiInstance.E_SetMotionVibraParam (3, E_SuppresLevel::LEVEL2);
```

6.2.92 获取震动抑制等级接口

功能介绍

用于查询机器人指定轴（1-6）的当前震动抑制等级。

参数说明

- axis：轴号，数据类型为 int，范围 1-6。

返回值

返回该轴的震动抑制等级，数据类型为 E_SuppresLevel。

配置示例

获取 3 轴的震动抑制等级，level 为 E_SuppresLevel::LEVEL2 表示当前是 2 级抑制。

```
E_SuppresLevel level = apiInstance.E_GetMotionVibraParam (3);
```

6.2.93 清除伺服报警接口

功能介绍

用于清除机器人指定配置名称、指定轴号的伺服报警，解除报警后机器人可恢复正常操作。

参数说明

- name：配置名称，数据类型为 ESString。
- index：配置序号（轴号），数据类型为 int。

返回值

清除成功返回 true，失败返回 false，数据类型为 bool。

配置示例

清除配置名称为 SERVO_CONFIG、轴号 1 的伺服报警；isSuccess 为 true 表示报警清除成功。

```
bool isSuccess = apiInstance.E_ResetServoError ("SERVO_CONFIG", 1);
```

6.2.94 设置伺服参数接口

功能介绍

用于修改机器人指定配置名称、指定轴号的伺服参数值。

参数说明

- name：配置名称，数据类型为 ESString。
- index：配置序号（轴号），数据类型为 int。
- value：参数值，数据类型为 int。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将配置名称为 POS_GAIN、轴号 2 的伺服参数值设置为 500；isSuccess 为 true 表示参数修改成功。

```
bool isSuccess = apiInstance.E_SetServoParam ("POS_GAIN", 2, 500);
```

6.2.95 获取伺服参数接口

功能介绍

用于查询机器人指定配置名称、指定轴号（1-16）的伺服参数值，通过引用参数返回。

参数说明

- name：配置名称，数据类型为 ESString。
- index：配置序号（轴号），数据类型为 int，范围 1-16。
- value：参数值返回值，数据类型为 int&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取配置名称为 POS_GAIN、轴号 2 的伺服参数值并存储在 paramValue 中；isSuccess 为 true 时 paramValue 为有效参数值。

```
int paramValue;bool isSuccess = apiInstance.E_GetServoParam ("POS_GAIN", 2, paramValue);
```

6.2.96 操作伺服参数接口

功能介绍

用于对机器人伺服参数执行批量备份或恢复操作，可选择操作出厂参数或调试参数。

参数说明

- opType：操作类型，数据类型为 int，1 表示备份，2 表示恢复。
- paramType：参数类型，数据类型为 int，1 表示出厂参数，2 表示调试参数。

返回值

操作成功返回 true，失败返回 false，数据类型为 bool。

配置示例

备份伺服调试参数（opType=1 备份，paramType=2 调试参数）；isSuccess 为 true 表示备份操作成功。

```
bool isSuccess = apiInstance.E_OpServoPnParam (1, 2);
```

6.2.97 伺服备份（C1 控制柜单轴）接口

功能介绍

针对 C1 控制柜机器人，对指定轴（或 IO）执行伺服固件 / 参数的备份操作。

参数说明

- index：轴号 / IO 标识，数据类型为 int，范围 [1-16]（轴号），0 表示 IO。
- fileType：文件类型，数据类型为 int，1 表示固件（fireware），2 表示参数（param）。

返回值

备份成功返回 true，失败返回 false，数据类型为 bool。

配置示例

备份 C1 控制柜机器人 3 轴的伺服参数（fileType=2）；isSuccess 为 true 表示备份完成。

```
bool isSuccess = apiInstance.E_ServoFoeUpload (3, 2);
```

6.2.98 伺服恢复（C1 控制柜单轴）接口

功能介绍

针对 C1 控制柜机器人，对指定轴（或 IO）执行伺服固件 / 参数的恢复操作，可选择恢复出厂参数或调试参数。

参数说明

- index：轴号 / IO 标识，数据类型为 int，范围 [1-16]（轴号），0 表示 IO。
- fileType：文件类型，数据类型为 int，1 表示固件，2 表示参数。
- paramType：参数类型，数据类型为 int，1 表示出厂参数，2 表示调试参数。

返回值

恢复成功返回 true，失败返回 false，数据类型为 bool。

配置示例

恢复 C1 控制柜机器人 3 轴的伺服出厂参数（fileType=2 参数，paramType=1 出厂参数）；isSuccess 为 true 表示恢复成功。

```
bool isSuccess = apiInstance.E_ServoFoeDownload (3, 2, 1);
```

6.2.99 获取伺服信息接口

功能介绍

用于查询机器人指定轴、指定配置名称对应的伺服版本信息。

参数说明

- name：配置名称，数据类型为 ESString。
- index：轴号，数据类型为 int，范围 [1-16]。
- value：版本信息返回值，数据类型为 ESString&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取 2 轴伺服版本信息并存储在 servoVersion 中；isSuccess 为 true 表示获取成功。

```
ESString servoVersion;
bool isSuccess = apiInstance.E_GetServoVersion ("SERVO_VERSION", 2, servoVersion);
```

6.2.100 获取伺服历史报警接口

功能介绍

用于查询机器人指定轴的伺服历史报警列表，最多返回 10 条报警记录。

参数说明

- index：轴号，数据类型为 int，范围 [1-16]。
- errList：告警列表返回数组，数据类型为 int [10]，存储告警码。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取 5 轴伺服历史告警列表，结果存储在 errList 数组中；isSuccess 为 true 表示获取成功，可遍历数组查看告警码。

```
int errList [10] = {0};
bool isSuccess = apiInstance.E_GetServoHisErrList (5, errList);
```

6.2.101 获取伺服状态接口

功能介绍

用于查询机器人指定轴的伺服实时状态，返回错误号、转速、扭矩等 6 项核心数据，需预先为数组申请内存。

参数说明

- index 轴号，数据类型为 int，范围 [1-16]。
- dataout 状态数据返回数组，数据类型为 double [6]，需调用者申请空间，数组元素依次为：错误号、转速、扭矩、过载比、电压、温度。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取 1 轴伺服状态，servoStatus [0] 为错误号、servoStatus [1] 为转速、servoStatus [2] 为扭矩等；isSuccess 为 true 表示数据有效。

```
double servoStatus [6] = {0.0};
bool isSuccess = apiInstance.E_GetServoStatus (1, servoStatus);
```

6.2.102 设置碰撞检测使能接口

功能介绍

用于启用或禁用机器人的碰撞检测功能，启用后机器人可检测运动过程中的碰撞并触发保护。

参数说明

- enable：使能开关，数据类型为 bool，true 表示使能，false 表示不使能。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

启用机器人碰撞检测功能；isSuccess 为 true 表示配置生效，机器人运动时可检测碰撞。

```
bool isSuccess = apiInstance.E_SetCollisionEnable (true);
```

6.2.103 获取碰撞检测使能状态接口

功能介绍

用于查询机器人碰撞检测功能的当前使能状态，通过引用参数返回结果。

参数说明

- enable：使能状态返回值，数据类型为 bool&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取碰撞检测使能状态，collisionEnable 为 true 表示功能已启用。

```
bool collisionEnable;
bool isSuccess = apiInstance.E_GetCollisionEnable (collisionEnable);
```

6.2.104 设置碰撞检测参数接口

功能介绍

用于为机器人指定轴（1-6）配置碰撞检测灵敏度，数值越小灵敏度越高（最大值 100%）。

参数说明

- axis：轴号，数据类型为 int，范围 1~6。
- perValue：检测灵敏度，数据类型为 int，数值越小越灵敏（最大 100）。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将 4 轴碰撞检测灵敏度设置为 50%；isSuccess 为 true 表示配置生效，灵敏度中等。

```
bool isSuccess = apiInstance.E_SetCollisionParam (4, 50);
```

6.2.105 获取碰撞检测参数接口

功能介绍

用于查询机器人指定轴（1-6）的碰撞检测灵敏度参数。

参数说明

- axis：轴号，数据类型为 int，范围 1~6。

返回值

返回检测灵敏度值，数据类型为 int；小于 0 表示获取失败，其他值为灵敏度（数值越小越灵敏，最大 100）。

配置示例

获取 4 轴碰撞检测灵敏度，sensValue 为 50 表示灵敏度 50%，为 - 1 表示获取失败。

```
int sensValue = apiInstance.E_GetCollisionParam (4);
```

6.2.106 单轴回零接口

功能介绍

用于控制机器人指定轴（1-6）执行单轴回零操作，需指定回零的单圈值。

参数说明

- axis：轴号，数据类型为 int，范围 1-6。
- value：单圈值，数据类型为 unsigned int。

返回值

回零成功返回 true，失败返回 false，数据类型为 bool。

配置示例

控制 2 轴以单圈值 1024 执行回零操作；isSuccess 为 true 表示回零完成。

```
bool isSuccess = apiInstance.E_HomeSingleJoint (2, 1024);
```

6.2.107 整体回零接口

功能介绍

用于控制机器人所有轴（1-6）执行整体回零操作，无需指定单圈值。

参数说明

无

返回值

回零成功返回 true，失败返回 false，数据类型为 bool。

配置示例

执行机器人整体回零操作；isSuccess 为 true 表示所有轴回零完成。

```
bool isSuccess = apiInstance.E_HomeAllJoint ();
```

6.2.108 获取单圈值接口

功能介绍

用于查询机器人指定轴（1-6）的单圈值，该值用于单轴回零配置。

参数说明

- axis：轴号，数据类型为 int，范围 1-6。

返回值

返回单圈值，数据类型为 unsigned int；0 表示获取失败。

配置示例

获取 2 轴单圈值，zeroValue 为 1024 表示有效，为 0 表示获取失败。

```
unsigned int zeroValue = apiInstance.E_GetEncoderValueOfZore (2);
```

6.2.109 设置机器人安装方式接口

功能介绍

用于配置机器人的安装方式（如地面安装、吊装、侧装等）。

参数说明

- InstallMode：安装类型，数据类型为 E_RobotInstallType。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将机器人安装方式设置为吊装；isSuccess 为 true 表示配置生效。

```
bool isSuccess = apiInstance.E_SetRobotInstallMode (E_RobotInstallType::CEILING);
```

6.2.110 获取机器人的安装方式接口

功能介绍

用于查询机器人当前的安装方式。

参数说明

无

返回值

返回安装类型，数据类型为 E_RobotInstallType。

配置示例

获取安装方式，installMode 为 E_RobotInstallType::GROUND 表示地面安装。

```
E_RobotInstallType installMode = apiInstance.E_GetRobotInstallMode ();
```

6.2.111 获取是否支持吊装接口

功能介绍

用于查询当前机器人型号是否支持吊装安装方式。

参数说明

无

返回值

支持吊装返回 true，不支持返回 false，数据类型为 bool。

配置示例

判断是否支持吊装，isSupportCeiling 为 true 表示支持，可配置吊装模式。

```
bool isSupportCeiling = apiInstance.E_GetRobotSupportCeiling ();
```

6.2.112 向机器人推送报警信息接口

功能介绍

用于向机器人控制器推送自定义报警信息，报警编号需使用 90000 以上的自定义值。

参数说明

- ErrInfo：告警信息，数据类型为 ESString。
- ErrNum：告警编号，数据类型为 int，需大于等于 90000。

返回值

推送成功返回 true，失败返回 false，数据类型为 bool。

配置示例

推送编号 90001 的自定义碰撞告警；isSuccess 为 true 表示推送成功。

```
bool isSuccess = apiInstance.E_SetRtToError ("自定义碰撞告警", 90001);
```

6.2.113 获取当前错误个数接口

功能介绍

用于查询机器人当前存在的错误 / 告警总数。

参数说明

无

返回值

返回错误个数，数据类型为 int；小于 0 表示获取失败。

配置示例

获取当前错误总数，errNum 为 3 表示有 3 个错误，为 - 1 表示获取失败。

```
int errNum = apiInstance.E_GetErrorNum ();
```

6.2.114 获取指定的错误信息接口

功能介绍

用于查询机器人指定序号的错误详情，包括告警编号、等级、内容、时间，需结合 E_GetErrorNum 使用。

参数说明

- index：错误序号，数据类型为 int，从 0 开始。
- code：告警编号返回值，数据类型为 int&。
- level：告警等级返回值，数据类型为 E_ErrorLevel&。
- text：告警内容返回值，数据类型为 ESString&。
- time：告警时间返回值，数据类型为 ESString&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取第 0 个错误的详情，errCode 为错误编号、errLevel 为错误等级、errText 为错误内容、errTime 为错误发生时间。

```
int errCode;
E_ErrorLevel errLevel;
ESString errText, errTime;
bool isSuccess = apiInstance.E_GetErrorInfo (0, errCode, errLevel, errText, errTime);
```

6.2.115 清除所有错误报警接口

功能介绍

用于清除机器人当前所有的错误 / 报警信息。

参数说明

无

返回值

清除成功返回 true，失败返回 false，数据类型为 bool。

配置示例

清除所有错误报警；isSuccess 为 true 表示清除完成。

```
bool isSuccess = apiInstance.E_ClearError ();
```

6.2.116 清除所有错误报警后获取是否还有错误接口

功能介绍

清除所有错误报警后，立即查询是否仍有未清除的错误。

参数说明

无

返回值

没有错误返回 true，有错误返回 false，数据类型为 bool。

配置示例

清除所有错误后检查状态，isClear 为 true 表示无残留错误，false 表示仍有错误。

```
bool isClear = apiInstance.E_CheckClearError ();
```

6.2.117 清除错误报警（指定错误号）接口

功能介绍

用于清除机器人指定错误号的单个错误 / 告警信息。

参数说明

- errorId：错误号，数据类型为 int。

返回值

清除成功返回 true，失败返回 false，数据类型为 bool。

配置示例

清除错误号 90001 的自定义告警；isSuccess 为 true 表示清除成功。

```
bool isSuccess = apiInstance.E_ClearError (90001);
```

6.2.118 获取机器人当前错误号（最近触发）接口

功能介绍

用于查询机器人最近触发的错误编号。

参数说明

无

返回值

返回最近触发的错误号，数据类型为 int。

配置示例

获取最近错误号，curErrId 为 90001 表示最近触发的是该编号的错误。

```
int curErrId = apiInstance.E_GetErrorId ();
```

6.2.119 获取机器人错误内容（默认语言）接口

功能介绍

用于查询指定错误号对应的错误详情，默认返回中文描述。

参数说明

- errorId：错误号，数据类型为 int。

返回值

返回错误详情，数据类型为 ESString。

配置示例

获取错误号 90001 的中文错误详情。

```
ESString errText = apiInstance.E_GetErrorString (90001);
```

6.2.120 获取机器人错误内容（指定语言）接口

功能介绍

用于查询指定错误号对应的错误详情，可指定返回中文（ch）或英文（en）描述。

参数说明

- errorId：错误号，数据类型为 int。
- language：语言类型，数据类型为 ESString，默认 "ch"（中文），可选 "en"（英文）。

返回值

返回对应语言的错误详情，数据类型为 ESString。

配置示例

获取错误号 90001 的英文错误详情。

```
ESString errText = apiInstance.E_GetErrorString (90001, "en");
```

6.2.121 获取当前位置信息（关节坐标系）接口

功能介绍

用于查询机器人当前在关节坐标系下的位置信息（关节角度）。

参数说明

无

返回值

返回位置信息，数据类型为 E_ROB_POS。

配置示例

获取关节坐标系下的当前位置，curJPos 包含各轴关节角度。

```
E_ROB_POS curJPos = apiInstance.E_GetCurJPos ();
```

6.2.122 获取当前位置信息（用户坐标系）接口

功能介绍

用于查询机器人当前在用户坐标系下的位置信息（笛卡尔坐标）。

参数说明

无

返回值

返回位置信息，数据类型为 E_ROB_POS

配置示例

获取用户坐标系下的当前位置，curWPos 包含 x/y/z/abc 坐标。

```
E_ROB_POS curWPos = apiInstance.E_GetCurWPos ();
```

6.2.123 获取当前位置信息（世界坐标系）接口

功能介绍

用于查询机器人当前在世界坐标系下的位置信息（笛卡尔坐标）。

参数说明

无

返回值

返回位置信息，数据类型为 E_ROB_POS。

配置示例

获取世界坐标系下的当前位置，curWorldPos 包含全局笛卡尔坐标。

```
E_ROB_POS curWorldPos = apiInstance.E_GetCurWorldPos ();
```

6.2.124 获取当前位置信息（用户坐标系，带时间戳）接口

功能介绍

用于查询机器人当前在用户坐标系下的位置信息，同时返回数据的时间戳。

参数说明

- timeStamp：时间戳返回指针，数据类型为 long*。

返回值

返回位置信息，数据类型为 E_ROB_POS。

配置示例

获取用户坐标系位置及时间戳，timeStamp 为数据采集的时间戳。

```
long timeStamp = 0;
E_ROB_POS curWPos = apiInstance.E_GetCurWPos2 (&timeStamp);
```

6.2.125 获取当前位置信息（世界坐标系，带时间戳）接口

功能介绍

用于查询机器人当前在世界坐标系下的位置信息，同时返回数据的时间戳。

参数说明

- timeStamp：时间戳返回指针，数据类型为 long*。

返回值

返回位置信息，数据类型为 E_ROB_POS。

配置示例

获取世界坐标系位置及时间戳，timeStamp 为数据采集的时间戳。

```
long timeStamp = 0;
E_ROB_POS curWorldPos = apiInstance.E_GetCurWorldPos2 (&timeStamp);
```

6.2.126 设置当前外部坐标系（按变量名）接口

功能介绍

用于通过外部坐标系变量名加载指定的外部坐标系。

参数说明

- ecpName：外部坐标系变量名，数据类型为 ESString，例："t_g.EXTTCP1"。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

加载变量名为 t_g.EXTTCP1 的外部坐标系；isSuccess 为 true 表示加载成功。

```
bool isSuccess = apiInstance.E_SetExternalTCPByName ("t_g.EXTTCP1");
```

6.2.127 设置当前外部坐标系（按 ID）接口

功能介绍

用于通过外部坐标系 ID 加载指定的外部坐标系，传入 0 表示不加载。

参数说明

- ecpid：外部坐标系 ID，数据类型为 int，0 表示不加载。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

加载 ID 为 1 的外部坐标系；传入 0 则卸载当前外部坐标系。

```
bool isSuccess = apiInstance.E_SetExternalTCPByID (1);
```

6.2.128 获取当前加载的程序名接口

功能介绍

用于查询机器人当前加载的工程程序名称。

参数说明

- value：工程程序名返回值，数据类型为 E_ROB_PROGNAME&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取当前加载的程序名，progName 存储工程 + 程序名称。

```
E_ROB_PROGNAME progName;
bool isSuccess = apiInstance.E_GetCurProgramName (progName);
```

6.2.129 获取当前程序运行状态接口

功能介绍

用于查询机器人当前加载程序的运行状态（运行、暂停、停止等）。

参数说明

无

返回值

返回程序运行状态，数据类型为 E_RunStatusType。

配置示例

获取程序状态，runStatus 为 E_RunStatusType::RUNNING 表示程序正在运行。

```
E_RunStatusType runStatus = apiInstance.E_GetCurProgramRunStatus ();
```

6.2.130 卸载工程接口

功能介绍

用于卸载机器人当前加载的工程文件。

参数说明

无

返回值

卸载成功返回 true，失败返回 false，数据类型为 bool。

配置示例

卸载当前工程；isSuccess 为 true 表示卸载完成。

```
bool isSuccess = apiInstance.E_UnLoadProject ();
```

6.2.131 卸载程序接口

功能介绍

用于卸载机器人当前加载的程序文件。

参数说明

无

返回值

卸载成功返回 true，失败返回 false，数据类型为 bool。

配置示例

卸载当前程序；isSuccess 为 true 表示卸载完成。

```
bool isSuccess = apiInstance.E_UnLoadProgram ();
```

6.2.132 只加载工程接口

功能介绍

用于加载指定名称的机器人工程文件（仅加载工程，不加载程序）。

参数说明

- prjname：工程名，数据类型为 ESString。

返回值

加载成功返回 true，失败返回 false，数据类型为 bool。

配置示例

加载名为 Welding_Project 的工程；isSuccess 为 true 表示加载成功。

```
bool isSuccess = apiInstance.E_LoadProject ("Welding_Project");
```

6.2.133 加载程序接口

功能介绍

用于加载指定工程下的指定程序文件。

参数说明

- prjname：工程名，数据类型为 ESString。
- programname：程序名，数据类型为 ESString。

返回值

加载成功返回 true，失败返回 false，数据类型为 bool。

配置示例

加载 Welding_Project 工程下的 Main_Prog 程序；isSuccess 为 true 表示加载成功。

```
bool isSuccess = apiInstance.E_LoadProgame ("Welding_Project", "Main_Prog");
```

6.2.134 刷新变量依赖接口

功能介绍

用于刷新机器人程序中变量的依赖关系，确保变量引用生效。

参数说明

无

返回值

刷新成功返回 true，失败返回 false，数据类型为 bool。

配置示例

刷新变量依赖；isSuccess 为 true 表示刷新完成。

```
bool isSuccess = apiInstance.E_RefrushVarCE ();
```

6.2.135 设置自启动程序接口

功能介绍

用于配置机器人开机自启动的工程和程序，重启后自动加载并运行。

参数说明

- prjname：工程名称，数据类型为 ESString。
- programname：程序名称，数据类型为 ESString。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

设置开机自启动 Welding_Project 工程下的 Main_Prog 程序；isSuccess 为 true 表示配置生效。

```
bool isSuccess = apiInstance.E_SetProgAutoload ("Welding_Project", "Main_Prog");
```

6.2.136 加载自动加载程序接口

功能介绍

加载示教器上预先配置的自动加载程序，需先在示教器完成自动加载程序的设置，接口仅执行加载操作。

参数说明

无

返回值

加载成功返回 true，失败返回 false，数据类型为 bool。

配置示例

加载示教器配置的自启动程序；isSuccess 为 true 表示程序加载完成。

```
bool isSuccess = apiInstance.E_LoadAutoloadProg ();
```

6.2.137 设置程序运行指针位置接口

功能介绍

指定程序运行的行号指针位置，使程序从该行开始执行。

参数说明

- lineNum：程序行号，数据类型为 int。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将程序运行指针设置到第 10 行；isSuccess 为 true 表示指针位置生效，后续运行程序从第 10 行开始。

```
bool isSuccess = apiInstance.E_SetProgPC (10);
```

6.2.138 获取程序指针接口

功能介绍

查询当前程序运行指针所在的行号。

参数说明

无

返回值

返回程序指针行号，数据类型为 int；小于 0 表示获取失败。

配置示例

获取当前程序指针行号，pcLine 为 10 表示指针在第 10 行，为 - 1 表示获取失败。

```
int pcLine = apiInstance.E_GetCurrPC ();
```

6.2.139 运行程序接口

功能介绍

启动当前加载的机器人程序，程序从当前指针位置开始执行。

参数说明

无

返回值

运行指令下发成功返回 true，失败返回 false，数据类型为 bool。

配置示例

启动程序运行；isSuccess 为 true 表示指令下发成功，程序开始执行。

```
bool isSuccess = apiInstance.E_ProgRun ();
```

6.2.140 暂停程序接口

功能介绍

暂停当前正在运行的机器人程序，暂停后程序指针保留在当前行。

参数说明

无

返回值

暂停成功返回 true，失败返回 false，数据类型为 bool。

配置示例

暂停运行中的程序；isSuccess 为 true 表示程序暂停，可通过 E_ProgRun 恢复运行。暂停运行中的程序；isSuccess 为 true 表示程序暂停，可通过 E_ProgRun 恢复运行。

```
bool isSuccess = apiInstance.E_ProgPause ();
```

6.2.141 停止程序接口

功能介绍

停止当前运行 / 暂停的机器人程序，停止后程序指针重置（通常回到首行）。

参数说明

无

返回值

停止成功返回 true，失败返回 false，数据类型为 bool。

配置示例

停止程序运行；isSuccess 为 true 表示程序停止，需重新设置指针或直接运行。

```
bool isSuccess = apiInstance.E_ProgStop ();
```

6.2.142 读取 Int 型变量值接口

功能介绍

读取指定名称、指定作用域的整型变量值。

参数说明

- varName：变量名称，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- valueBack：变量值返回值，数据类型为 int&（引用类型）。

返回值

读取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取全局作用域下整型变量 g_count 的值，结果存储在 intValue 中；isSuccess 为 true 表示读取成功。

```
int intValue;
bool isSuccess = apiInstance.E_ReadInt ("g_count", E_ScopeType::GLOBAL, intValue);
```

6.2.143 写入 Int 型变量值接口

功能介绍

修改指定名称、指定作用域的整型变量值。

参数说明

- varName：变量名称，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- value：变量值，数据类型为 int。

返回值

写入成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将全局变量 g_count 的值设置为 100；isSuccess 为 true 表示变量值修改成功。

```
bool isSuccess = apiInstance.E_WriteInt ("g_count", E_ScopeType::GLOBAL, 100);
```

6.2.144 读取 Double 型变量值接口

功能介绍

读取指定名称、指定作用域的浮点型（Double）变量值。

参数说明

- varName：变量名称，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- valueBack：变量值返回值，数据类型为 double&（引用类型）。

返回值

读取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取局部作用域下浮点型变量 g_speed 的值；isSuccess 为 true 时 realValue 为有效数值。

```
double realValue;
bool isSuccess = apiInstance.E_ReadReal ("g_speed", E_ScopeType::LOCAL, realValue);
```

6.2.145 写入 Double 型变量值接口

功能介绍

修改指定名称、指定作用域的浮点型（Double）变量值。

参数说明

- varName：变量名称，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- value：变量值，数据类型为 double。

返回值

写入成功返回 true，失败返回 false，数据类型为 bool

配置示例

将局部变量 g_speed 的值设置为 50.5；isSuccess 为 true 表示修改生效。

```
bool isSuccess = apiInstance.E_WriteReal ("g_speed", E_ScopeType::LOCAL, 50.5);
```

6.2.146 读取 String 型变量值接口

功能介绍

读取指定名称、指定作用域的字符串型变量值。

参数说明

- varName：变量名称，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- valueBack：变量值返回值，数据类型为 ESString&（引用类型）。

返回值

读取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取全局字符串变量 g_msg 的值；isSuccess 为 true 时 strValue 为有效字符串。

```
ESString strValue;
bool isSuccess = apiInstance.E_ReadStr ("g_msg", E_ScopeType::GLOBAL, strValue);
```

6.2.147 写入 String 型变量值接口

功能介绍

修改指定名称、指定作用域的字符串型变量值。

参数说明

- varName：变量名称，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- value：变量值，数据类型为 ESString。

返回值

写入成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将全局变量 g_msg 的值设置为 "运动完成"；isSuccess 为 true 表示修改成功。

```
bool isSuccess = apiInstance.E_WriteStr ("g_msg", E_ScopeType::GLOBAL, "运动完成");
```

6.2.148 读取位置类型变量值接口

功能介绍

读取指定名称、指定作用域的位置类型（E_ROB_POS）变量值，包含坐标 / 关节角度信息。

参数说明

- varName：变量名称，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- valueBack：变量值返回值，数据类型为 E_ROB_POS&（引用类型）。

返回值

读取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取局部位置变量 p_home 的值；isSuccess 为 true 时 posValue 包含完整的位置坐标。

```
E_ROB_POS posValue;
bool isSuccess = apiInstance.E_ReadPos ("p_home", E_ScopeType::LOCAL, posValue);
```

6.2.149 写入位置类型变量值接口（E_WritePos/E_WritePos2）

功能介绍

修改指定名称、指定作用域的位置类型（E_ROB_POS）变量值，两个接口功能一致（E_-WritePos2 为兼容扩展）。

参数说明

- varName：变量名称，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- value：变量值，数据类型为 E_ROB_POS。

返回值

写入成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将局部位置变量 p_home 的值设置为 newPos；E_WritePos2 使用方式完全相同。

```
E  ROB  POS newPos;
bool isSuccess = apiInstance.E_WritePos ("p_home", E_ScopeType::LOCAL, newPos);
```

6.2.150 设置 DO 的值接口

功能介绍

批量设置多个数字输出（DO）的状态，通过数组元素的正负标识 IO 状态（+ 为 1 / 开启，-为 0 / 关闭），数值为 IO 下标。

参数说明

- IoValues IO：值数组指针，数据类型为 int*。
- number：数组元素个数，数据类型为 int。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

批量设置 4 个 DO 的状态；isSuccess 为 true 表示所有 DO 状态设置生效。

```
设置成功返回 true，失败返回 false，数据类型为 bool。
```

6.2.151 获取最近一次下发成功的运动指令的信息接口

功能介绍

查询最近一次成功下发的运动指令的唯一标识（JobId）。

参数说明

返回值

返回运动指令信息，数据类型为 E_ROB_JOBID。

配置示例

获取最近运动指令的 JobId，用于后续查询该指令的执行状态。

```
E_ROB_JOBID curJobId = apiInstance.E_GetCurMovCmdJobId ();
```

6.2.152 获取最近一次下发成功的运动指令的执行状态接口

功能介绍

查询最近一次运动指令的执行状态，返回值对应不同状态码。

参数说明

无

返回值

返回状态码，数据类型为 int：-2 = 错误、-1 = 未找到、0 = 运动未完成、1 = 运动完成、2 = 暂停、3 = 结束 / 运动错误、4 = 预留。

配置示例

cmdStatus 为 1 表示最近运动指令已完成，为 0 表示仍在运动中。

```
int cmdStatus = apiInstance.E_GetCurMovCmdSts ();
```

6.2.153 获取指定运动指令执行状态接口

功能介绍

根据运动指令的 JobId，查询该指令的具体执行状态。

参数说明

- jobID：指定的运动指令标识，数据类型为 E_ROB_JOBID。

返回值

返回状态码，数据类型为 int（状态码含义同 E_GetCurMovCmdSts）。

配置示例

查询 JobId 为 curJobId 的运动指令状态；可用于批量管理多个运动指令。

```
int cmdStatus = apiInstance.E_GetMovCmdSts (curJobId);
```

6.2.154 接口耗时接口

功能介绍

查询上一次调用接口的执行耗时（单位通常为秒 / 毫秒，需结合实际 SDK 定义）。

参数说明

无

返回值

返回接口耗时，数据类型为 double。

配置示例

获取上一次接口的执行耗时，用于性能监控和调试。

```
double execTime = apiInstance.getLastFuncExecTime ();
```

6.2.155 设置点位数组接口

功能介绍

批量设置位置数组变量的元素值，仅支持 CposArray（笛卡尔坐标数组）或 AposArray（关节坐标数组）类型。

参数说明

- varType：变量类型，数据类型为 E_VarType。
- varName：变量名，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- startIndex：起始下标，数据类型为 int。
- nums：存储个数，数据类型为 int。
- dataIn：点位数组指针，数据类型为 E_ROB_POS*。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

从全局位置数组 pos_list 的第 0 个下标开始，写入 3 个笛卡尔坐标值。

```
E_ROB_POS posArray [3];
bool isSuccess = apiInstance.E_SetPosArrayValue (E_VarType::CposArray, "pos_list",
E_ScopeType::GLOBAL, 0, 3, posArray);
```

6.2.156 获取点位数组接口

功能介绍

批量读取位置数组变量的元素值，仅支持 CposArray/AposArray 类型，需预先为返回数组申请内存。

参数说明

- varType：变量类型，数据类型为 E_VarType。
- varName：变量名，数据类型为 ESString。
- varScope：变量域，数据类型为 E_ScopeType。
- startIndex：起始下标，数据类型为 int。
- nums：读取个数，数据类型为 int。
- dataOut：点位数组返回指针，数据类型为 E_ROB_POS*。

返回值

读取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取全局关节坐标数组 pos_list 中从第 0 个下标开始的 3 个值。

```
E_ROB_POS posArray [3];
bool isSuccess = apiInstance.E  GetPosArrayValue (E  VarType::AposArray, "pos  list",
E_ScopeType::GLOBAL, 0, 3, posArray);
```

6.2.157 获取力矩的值接口

功能介绍

读取机器人各轴的力矩值，返回 16 个 double 类型的力矩数据（覆盖本体轴 + 附加轴），需预先申请内存。

参数说明

- dataOut：力矩值返回数组指针，数据类型为 double*（需申请 16 个 double 的内存）。

返回值

读取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取 16 个轴的力矩值，存储在 torqueData 数组中；isSuccess 为 true 表示数据有效。

```
double torqueData [16] = {0.0};
bool isSuccess = apiInstance.E_GetAxeTorqueValue (torqueData);
```

6.2.158 设置控制器网络配置接口

功能介绍

配置机器人控制器 lan3、lan4 网口的 IP、子网掩码、网关信息。

参数说明

- sRtosLan3IP：lan3 口 IP，数据类型为 ESString。
- sRtosLan3IPNetmask：lan3 口掩码，数据类型为 ESString。
- sRtosLan3IPGateway：lan3 口网关，数据类型为 ESString。
- sContrLan4IP：lan4 口 IP，数据类型为 ESString。
- sContrLan4IPNetmask：lan4 口掩码，数据类型为 ESString。
- sContrLan4IPGateway：lan4 口网关，数据类型为 ESString。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

配置 lan3 和 lan4 的网络参数；isSuccess 为 true 表示配置生效（通常需重启控制器）。

```
bool isSuccess = apiInstance.E_SetControllerExtIP ("192.168.3.100", "255.255.255.0",
"192.168.3.1","192.168.4.100", "255.255.255.0", "192.168.4.1");
```

6.2.159 获取控制器网络配置接口

功能介绍

查询机器人控制器 lan3、lan4 网口的 IP、子网掩码、网关信息。

参数说明

- sRtosLan3IP：lan3 口 IP 返回值，数据类型为 ESString&。
- sRtosLan3IPNetmask：lan3 口掩码返回值，数据类型为 ESString&。
- sRtosLan3IPGateway：lan3 口网关返回值，数据类型为 ESString&。
- sContrLan4IP：lan4 口 IP 返回值，数据类型为 ESString&。
- sContrLan4IPNetmask：lan4 口掩码返回值，数据类型为 ESString&。
- sContrLan4IPGateway：lan4 口网关返回值，数据类型为 ESString&。

返回值

读取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取所有网口的网络配置；isSuccess 为 true 时各变量存储对应参数。

```
ESString lan3IP, lan3Mask, lan3Gateway, lan4IP, lan4Mask, lan4Gateway;
bool isSuccess = apiInstance.E  GetControllerExtIP (lan3IP, lan3Mask, lan3Gateway,
lan4IP, lan4Mask, lan4Gateway);
```

6.2.160 获取 IO 个数接口

功能介绍

查询指定类型 IO 的总数量。

参数说明

- IoType：IO 类型，数据类型为 E_IOType。

返回值

返回 IO 个数，数据类型为 int；小于 0 表示获取失败。

配置示例

获取数字输出 IO 的总数，doCount 为 16 表示有 16 个 DO，为 - 1 表示获取失败。

```
int doCount = apiInstance.E_GetIOCount (E_IOType::DIGITAL_OUTPUT);
```

6.2.161 计算用户坐标系接口

功能介绍

根据 3 个参考点位、轴类型 / 方向、面类型 / 方向，自动计算用户坐标系参数。

参数说明

- coord：计算结果返回值，数据类型为 E_ROB_COORD&。
- UserCorrdinateParam：3 个参考点位数组，数据类型为 E_ROB_POS [3]。
- AxisType：轴类型，数据类型为 E_CoordAxisType。
- AxisDir：轴方向，数据类型为 E_DistentionType。
- PlaneType：面类型，数据类型为 E_CoordPlaneType。
- PlaneDir：面方向，数据类型为 E_DistentionType。

返回值

计算成功返回 true，失败返回 false，数据类型为 bool

配置示例

根据参考点计算用户坐标系，结果存储在 userCoord 中。

```
E_ROB_COORD userCoord;
E_ROB_POS refPoints [3];
bool isSuccess = apiInstance.E_CalculateUserCorrdinate (userCoord,
refPoints, E_CoordAxisType::X_AXIS,
E_DistentionType::POSITIVE,E_CoordPlaneType::XY_PLANE, E_DistentionType::POSITIVE);
```

6.2.162 计算工具坐标系_4 点法接口

功能介绍

通过 4 个参考点位，使用 4 点法计算工具坐标系参数，支持返回计算误差（两个重载版本，一个返回误差，一个不返回）。

参数说明

- coord：计算结果返回值，数据类型为 E_ROB_COORD&。
- errValue：计算误差返回值（仅第一个版本），数据类型为 double&。
- ToolCorrdinateParam：4 个参考点位数组，数据类型为 E_ROB_POS [4]。

返回值

计算成功返回 true，失败返回 false，数据类型为 bool。

配置示例

4 点法计算工具坐标系，err 存储计算误差，越小表示精度越高。

```
E_ROB_COORD toolCoord;
double err;
E_ROB_POS refPoints [4];
bool isSuccess1 = apiInstance.E_CalculateToolCorrdinate_4Pnt (toolCoord, err, refPoints);
bool isSuccess2 = apiInstance.E_CalculateToolCorrdinate_4Pnt (toolCoord, refPoints);
```

6.2.163 计算工具坐标系_1 点法接口

功能介绍

通过 1 个参考点位，使用 1 点法快速计算工具坐标系参数（精度低于 4 点法）。

参数说明

- coord：计算结果返回值，数据类型为 E_ROB_COORD&。
- ToolCorrdinateParam：参考点位，数据类型为 const E_ROB_POS&。

返回值

计算成功返回 true，失败返回 false，数据类型为 bool。

配置示例

1 点法计算工具坐标系，适用于精度要求较低的场景。

```
E_ROB_COORD toolCoord;E_ROB_POS refPoint;
bool isSuccess = apiInstance.E_CalculateToolCorrdinate_1Pnt (toolCoord, refPoint);
```

6.2.164 执行 MovJ2 数组指令接口

功能介绍

执行关节插补（MovJ）轨迹运动，支持多目标点位数组，可配置速度、过渡参数、负载 / 工具 / 用户坐标等，最大支持 50 个轨迹点。

参数说明

- dstPointArray：目标位置点数组，数据类型为 E_ROB_POS*。
- posNums：轨迹点个数（≤50），数据类型为 int。
- velo：运行速度，数据类型为 const E_ROB_SPEED&（默认值）。
- zone：过渡参数，数据类型为 const E_ROB_ZONE&（默认值）。
- payloadId：负载号（默认 0），数据类型为 int。
- toolId：工具号（默认 0），数据类型为 int。
- coordId：用户坐标号（默认 0），数据类型为 int。
- addDo：附加指令（默认空），数据类型为 const E_ROB_APIBUFF&。
- valfine：到位精度值（默认 0.0），数据类型为 double。
- IsWaitFinish：是否等待运动完成（默认 false），数据类型为 bool。

返回值

返回指令标识 JobId，数据类型为 E_ROB_JOBID；id<0 表示失败。

配置示例

执行 3 个点的关节插补运动，工具号 1，到位精度 0.05mm，不等待完成立即返回。

```
E_ROB_POS posArray [3];
E_ROB_SPEED speed;
speed.linear = 100;
E_ROB_JOBID jobId = apiInstance.E_MovJ2_Trajectory (posArray, 3,
speed, E_ROB_ZONE (), 0, 1, 0, E_ROB_APIBUFF (), 0.05, false);
```

6.2.165 执行 MovJOffset 指令接口

功能介绍

执行基于参考坐标系的关节插补偏移运动，从目标点按相对位置偏移运动。

参数说明

- dstPoint：目标点位，数据类型为 E_ROB_POS&。
- relativePoint：相对位置，数据类型为 E_ROB_POS&。
- RefSys：参考坐标系（0 = 用户坐标，1 = 工具坐标），数据类型为 int。
- velo：运行速度（默认值），数据类型为 const E_ROB_SPEED&。
- zone：过渡参数（默认值），数据类型为 const E_ROB_ZONE&。
- payloadId：负载号（默认 0），数据类型为 int。
- toolId：工具号（默认 0），数据类型为 int。
- coordId：用户坐标号（默认 0），数据类型为 int。
- addDo：附加指令（默认空），数据类型为 const E_ROB_APIBUFF&。
- valfine：到位精度值（默认 0.0），数据类型为 double。
- IsWaitFinish：是否等待完成（默认 false），数据类型为 bool。

返回值

返回指令标识 JobId，数据类型为 E_ROB_JOBID；id<0 表示失败。

配置示例

以用户坐标系为参考，从 dstPos 向 X 方向偏移 10mm 做关节插补运动。

```
E_ROB_POS dstPos, offsetPos;
offsetPos.x = 10;
E_ROB_JOBID jobId = apiInstance.E_MovJOffset (dstPos, offsetPos, 0, E_ROB_SPEED (), E_ROB_ZONE ());
```

6.2.166 执行MovJ2 数组指令（独立配置速度 / 过渡参数）接口

功能介绍

执行关节插补轨迹运动，轨迹点数组中每个点独立配置速度、过渡参数等，最大支持 500 个轨迹点。

参数说明

- dstPointArray：轨迹点数组（含位置、速度、过渡参数等），数据类型为 Trajectory2Pos*。
- posNums：轨迹点个数（≤500），数据类型为 int。
- addDo：附加指令（默认空），数据类型为 const E_ROB_APIBUFF&。
- valfine：到位精度值（默认 0.0），数据类型为 double。
- IsWaitFinish：是否等待完成（默认 false），数据类型为 bool。

返回值

返回指令标识 JobId，数据类型为 E_ROB_JOBID；id<0 表示失败。

配置示例

执行 2 个自定义参数的关节插补轨迹运动，支持更多轨迹点和灵活配置。

```
Trajectory2Pos posArray [2];
E_ROB_JOBID jobId = apiInstance.E_MovJ2_Trajectory2 (posArray, 2);
```

6.2.167 执行 MovL2 指令数组接口

功能介绍

执行线性插补（MovL）轨迹运动，支持多目标点位数组，最大支持 50 个轨迹点，可配置速率类型。

参数说明

- dstPointArray：目标位置点数组，数据类型为 E_ROB_POS*。
- posNums：轨迹点个数（≤50），数据类型为 int。
- velo：运行速度（默认值），数据类型为 const E_ROB_SPEED&。
- zone：过渡参数（默认值），数据类型为 const E_ROB_ZONE&。
- payloadId：负载号（默认 0），数据类型为 int。
- toolId：工具号（默认 0），数据类型为 int。
- coordId：用户坐标号（默认 0），数据类型为 int。
- overrideType：速率类型（0=GOVROFF，1=GOVRON，默认 1），数据类型为 int。
- addDo：附加指令（默认空），数据类型为 const E_ROB_APIBUFF&。
- valfine：到位精度值（默认 0.0），数据类型为 double。
- IsWaitFinish：是否等待完成（默认 false），数据类型为 bool。

返回值

返回指令标识 JobId，数据类型为 E_ROB_JOBID；id<0 表示失败。

配置示例

执行 3 个点的线性插补运动，速率类型为 GOVRON。

```
E_ROB_POS posArray [3];
E_ROB_SPEED speed;
speed.linear = 50;
```

```
E_ROB_JOBID jobId = apiInstance.E_MovL2_Trajectory (posArray, 3, speed,
 E_ROB_ZONE (), 0, 1, 0, 1, E_ROB_APIBUFF (), 0.05);
```

6.2.168 执行MovLOffset 指令接口

功能介绍

执行基于参考坐标系的线性插补偏移运动，从目标点按相对位置偏移运动。

参数说明

- dstPoint：目标位置点，数据类型为 const E_ROB_POS&。
- relativePoint：相对位置，数据类型为 const E_ROB_POS&。
- RefSys：参考坐标系（0 = 用户坐标，1 = 工具坐标），数据类型为 int。
- velo：运行速度（默认值），数据类型为 const E_ROB_SPEED&。
- zone：过渡参数（默认值），数据类型为 const E_ROB_ZONE&。
- payloadId：负载号（默认 0），数据类型为 int。
- toolId：工具号（默认 0），数据类型为 int。
- coordId：用户坐标号（默认 0），数据类型为 int。
- overrideType：速率类型（默认 1），数据类型为 int。
- addDo：附加指令（默认空），数据类型为 const E_ROB_APIBUFF&。
- valfine：到位精度值（默认 0.0），数据类型为 double。
- IsWaitFinish：是否等待完成（默认 false），数据类型为 bool。

返回值

返回指令标识 JobId，数据类型为 E_ROB_JOBID；id<0 表示失败。

配置示例

以工具坐标系为参考，从 dstPos 向 Z 负方向偏移 5mm 做线性插补运动。

```
E_ROB_POS dstPos, offsetPos;offsetPos.z = -5;
E_ROB_JOBID jobId = apiInstance.E_MovLOffset (dstPos, offsetPos, 1);
```

6.2.169 执行MovL2 指令数组（独立配置参数）接口

功能介绍

执行线性插补轨迹运动，轨迹点数组中每个点独立配置速度、过渡参数等，最大支持 500 个轨迹点。

参数说明

- dstPointArray 轨迹点数组（含位置、速度等参数），数据类型为 Trajectory2Pos*；posNums 轨迹点个数（≤500），数据类型为 int。
- overrideType 速率类型（默认 1），数据类型为 int。
- addDo 附加指令（默认空），数据类型为 const E_ROB_APIBUFF&。
- valfine 到位精度值（默认 0.0），数据类型为 double。
- IsWaitFinish 是否等待完成（默认 false），数据类型为 bool。

返回值

返回指令标识 JobId，数据类型为 E_ROB_JOBID；id<0 表示失败。

配置示例

执行 5 个点的线性插补运动，支持 500 个点的大容量轨迹。

```
Trajectory2Pos posArray [5];
E_ROB_JOBID jobId = apiInstance.E_MovL2_Trajectory2 (posArray, 5);
```

6.2.170 执行 MovJ3 运动指令接口

功能介绍

简化版关节插补运动指令，直接传入 X/Y/Z/A/B/C 坐标值，无需构造 E_ROB_POS 结构体，参数更简洁。

参数说明

- xValue/yValue/zValue/aValue/bValue/cValue：目标位置坐标，数据类型为 double。
- mode：位置模式（默认 - 1 = 当前模式），数据类型为 int。
- speedValue：速度值（默认 - 1=V100，范围 (0,4000]），数据类型为 int。
- zoneValue：过渡值（默认 - 1=C0，范围 [0,200]），数据类型为 int。
- IsWaitFinish：是否等待完成（默认 false），数据类型为 bool。

返回值

返回指令 JobId，数据类型为 int；id<0 表示失败。

配置示例

运动到 (100,50,80,0,90,0) 位置，速度 V200，过渡值 C10，不等待完成。

```
int jobId = apiInstance.E_MovJ3 (100, 50, 80, 0, 90, 0, -1, 200, 10, false);
```

6.2.171 执行 MovL3 运动指令接口

功能介绍

简化版线性插补运动指令，直接传入坐标值，参数简洁，使用方式与 MovJ3 一致。

参数说明

- xValue/yValue/zValue/aValue/bValue/cValue：目标位置坐标，数据类型为 double。
- mode：位置模式（默认 - 1），数据类型为 int。
- speedValue：速度值（默认 - 1），数据类型为 int。
- zoneValue：过渡值（默认 - 1），数据类型为 int。
- IsWaitFinish：是否等待完成（默认 false），数据类型为 bool。

返回值

返回指令 JobId，数据类型为 int；id<0 表示失败。

配置示例

使用默认参数（速度 V100、过渡值 C0）线性运动到目标位置。

```
int jobId = apiInstance.E_MovL3 (100, 50, 80, 0, 90, 0);
```

6.2.172 获取指定运动指令执行状态（MovJ3/MovL3 专用）接口

功能介绍

专门用于查询E_MovJ3/E_MovL3指令返回的运动 ID 对应的执行状态，状态码定义与通用版本一致。

参数说明

- jobID：运动 ID 号（MovJ3/MovL3 的返回值），数据类型为 int。

返回值

返回状态码，数据类型为 int：-2 = 错误、-1 = 未找到、0 = 运动未完成、1 = 运动完成、2 = 暂停、3 = 结束 / 运动错误、4 = 预留。

配置示例

查询 MovJ3 指令的执行状态，status 为 1 表示运动完成，为 0 表示仍在运动中。

```
int jobId = apiInstance.E_MovJ3 (100,50,80,0,90,0);
JobIdint status = apiInstance.E_GetMovCmdSts3 (jobId);
```

6.2.173 获取运动队列中尚未完成的运动段数量接口

功能介绍

查询机器人运动队列中未执行完成的运动段总数，同时反馈机器人当前运行模式（暂停 / 停止 / 通信错误）。

参数说明

无

返回值

返回数量 / 状态码，数据类型为 int：-3 = 机器人暂停、-2 = 机器人停止、-1 = 通信错误、≥0 = 未完成的运动段数量。

配置示例

unFinishNums 为 5 表示有 5 个未完成的运动段；为 - 3 表示机器人处于暂停状态。

```
int unFinishNums = apiInstance.E_GetUnFinishedJobNums ();
```

6.2.174 获取指定运动数组指令执行状态接口

功能介绍

查询轨迹数组类运动指令（如 MovJ2_Trajectory、MovL2_Trajectory）返回的 JobId 对应的执行状态。

参数说明

- posArrayJobID：轨迹指令标识，数据类型为 E_ROB_JOBID。

返回值

返回状态码，数据类型为 int（状态码含义同 E_GetMovCmdSts3）。

配置示例

查询轨迹数组指令的执行状态，可判断整段轨迹是否完成。

```
E_ROB_JOBID trajJobId = apiInstance.E_MovJ2_Trajectory(posArray, 3);
int status = apiInstance.E_GetTrajectoryCmdSts(trajJobId);
```

6.2.175 获取指定运动数组指令的子运动 ID接口

功能介绍

获取轨迹数组指令中每个子运动点对应的独立 JobId，便于逐个监控子运动的执行状态，数组最大支持 50 个元素。

参数说明

- posArrayJobID：轨迹指令标识，数据类型为 E_ROB_JOBID。
- posAllJobIDBuff：子运动 ID 返回数组指针，数据类型为 E_ROB_JOBID*。
- buffNums：数组容量（最大 50），数据类型为 int。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

获取轨迹指令中所有子运动的 JobId，存储在 subJobIds 数组中；isSuccess 为 true 时可遍历数组查询每个子运动状态。

```
E_ROB_JOBID subJobIds[50] = {0};
bool isSuccess = apiInstance.E_GetTrajectoryPointsJobId(trajJobId, subJobIds, 50);
```

6.2.176 笛卡尔坐标转关节坐标接口

功能介绍

将笛卡尔坐标系下的位置（含 mode、姿态、坐标）转换为关节坐标系下的各轴角度值。

参数说明

- Cpos：输入笛卡尔坐标数组（长度 13），数据类型为 double[13]，元素依次为：mode、cf1、cf2、cf3、cf4、cf5、cf6、x、y、z、a、b、c；Apos 输出关节坐标数组（长度 6），数据类型为 double[6]，存储 a1~a6 轴角度。

返回值

转换成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将笛卡尔坐标转换为关节角度，apos 数组存储 a1~a6 的角度值；isSuccess 为 true 表示转换有效。

```
double cpos [13] = {0,0,0,0,0,0,0, 100,50,80, 0,90,0};
double apos [6] = {0.0};
bool isSuccess = apiInstance.E_CposToAPos (cpos, apos);
```

6.2.177 关节坐标转笛卡尔坐标接口

功能介绍

将关节坐标系下的各轴角度值转换为笛卡尔坐标系下的位置（含 mode、姿态、坐标）。

参数说明

- Apos：输入关节坐标数组（长度 6），数据类型为 double [6]，存储 a1~a6 轴角度；Cpos 输出笛卡尔坐标数组（长度 13），数据类型为 double [13]（元素含义同 E_Cpos-ToAPos）。

返回值

转换成功返回 true，失败返回 false，数据类型为 bool。

配置示例

将关节角度转换为笛卡尔坐标，cpos 数组包含完整的位置信息。

```
double apos [6] = {0.0, 90.0, 0.0, 0.0, 90.0, 0.0};
double cpos [13] = {0.0};
bool isSuccess = apiInstance.E_AposToCPos (apos, cpos);
```

6.2.178 获取机器人当前线速度接口

功能介绍

实时查询机器人 TCP（工具中心点）的当前线速度值。

参数说明

无

返回值

返回线速度值，数据类型为 int（单位通常为 mm/s）。

配置示例

curVelo 为 200 表示当前 TCP 线速度为 200mm/s，可用于运动速度监控。

```
int curVelo = apiInstance.E_GetCurTcpVelo ();
```

6.2.179 弧焊通用协议设置指令接口

功能介绍

查询机器人当前配置的弧焊自定义协议内容。

参数说明

- procotol：协议内容返回值，数据类型为 ESString&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取当前弧焊协议配置，weldProtocol 存储完整的协议字符串。

```
ESString weldProtocol;
bool isSuccess = apiInstance.E_GetChannelForWeld (weldProtocol);
```

6.2.180 弧焊通用协议获取指令接口

功能介绍

查询机器人当前配置的弧焊自定义协议内容。

参数说明

- procotol：协议内容返回值，数据类型为 ESString&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取当前弧焊协议配置，weldProtocol 存储完整的协议字符串。

```
ESString weldProtocol;
bool isSuccess = apiInstance.E_GetChannelForWeld (weldProtocol);
```

6.2.181 弧焊状态获取指令接口

功能介绍

查询弧焊设备的实时状态信息（如起弧状态、电流、电压等）。

参数说明

- procotol：协议内容返回值，数据类型为 ESString&。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取弧焊状态，weldStatus 可能包含 "ARC_ON=1;CURRENT=200;VOLTAGE=28" 等信息。

```
ESString weldStatus;
bool isSuccess = apiInstance.E_GetWeldStatus (weldStatus);
```

6.2.182 设置世界坐标系加减速度接口

功能介绍

配置机器人在世界坐标系下运动的加速度、减速度、加加速度（Jerk），以及姿态相关的加减速参数，影响笛卡尔空间运动的平滑性。

参数说明

- acc：线加速度，数据类型为 double。
- dec：线减速度，数据类型为 double。
- jerk：线加加速度，数据类型为 double。
- oriacc：姿态加速度，数据类型为 double。
- oridec：姿态减速度，数据类型为 double。
- oridec：姿态减速度，数据类型为 double。
- orijerk：姿态加加速度，数据类型为 double。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

设置线加减速 500mm/s²、线加加速度 1000mm/s³，姿态加减速 300°/s²、姿态加加速度 600°/s³。

```
bool isSuccess = apiInstance.E_reset_CartDyn_RobotFace (500, 500, 1000, 300, 300, 600);
```

6.2.183 设置关节坐标系加减速度接口

功能介绍

配置机器人在关节坐标系下运动的加速度、减速度、加加速度，影响关节运动的平滑性和响应速度。

参数说明

- acc：关节加速度，数据类型为 double。
- dec：关节减速度，数据类型为 double。
- jerk：关节加加速度，数据类型为 double。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

设置关节加减速 180°/s²、加加速度 360°/s³，提升关节运动的平稳性。

```
bool isSuccess = apiInstance.E_reset_JointDyn_RobotFace (180, 180, 360);
```

6.2.184 设置轴运动参数接口

功能介绍

修改指定轴的运动参数（速度、加速度、加加速度），参数存储在 configfiles 目录下的 EAxisParam 文件中，需重启生效。

参数说明

- Axle_name：轴号，数据类型为 int。
- velo：速度值，数据类型为 int。
- acc：加速度值，数据类型为 int
- jerk：加加速度值，数据类型为 int。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

设置 1 轴的速度 300°/s、加速度 180°/s²、加加速度 360°/s³。

```
bool isSuccess = apiInstance.E_set_Robot_Motion_Param (1, 300, 180, 360);
```

6.2.185 获取轴运动参数接口

功能介绍

查询指定轴的运动参数（速度、加速度、加加速度），读取 EAxisParam 文件中的配置。

参数说明

- Axle_name：轴号，数据类型为 int。
- velo：速度值，数据类型为 int。
- acc：加速度值，数据类型为 int
- jerk：加加速度值，数据类型为 int。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取 1 轴的运动参数，velo/acc/jerk 存储对应数值。

```
int velo, acc, jerk;
bool isSuccess = apiInstance.E_get_Robot_Motion_Param (1, velo, acc, jerk);
```

6.2.186 设置轴限位参数接口

功能介绍

修改指定轴的正负限位值，参数存储在 configfiles 目录下的 robotparam 文件中，需重启生效。

参数说明

- Axle_name：轴号，数据类型为 int。
- profit：正限位值，数据类型为 int。
- loss：负限位值，数据类型为 int。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

设置 1 轴的正限位 180°、负限位 - 180°，限制轴的运动范围。

```
bool isSuccess = apiInstance.E_set_Robot_Limit_Param (1, 180, -180);
```

6.2.187 获取轴限位参数接口

功能介绍

查询指定轴的正负限位值，读取 robotparam 文件中的配置。

参数说明

- Axle_name：轴号，数据类型为 int。
- profit：正限位值，数据类型为 int。
- loss：负限位值，数据类型为 int。

返回值

获取成功返回 true，失败返回 false，数据类型为 bool。

配置示例

读取 1 轴的限位参数，profit=180、loss=-180 表示限位范围为 - 180°~180°。

```
int profit, loss;
bool isSuccess = apiInstance.E_get_Robot_Limit_Param (1, profit, loss);
```

6.2.188 获取机器人 Jog 状态接口

功能介绍

查询机器人当前是否处于手动 Jog 模式（点动模式）。

参数说明

无

返回值

处于 Jog 模式返回 true，否则返回 false，数据类型为 bool。

配置示例

isJogMode 为 true 表示机器人当前是手动点动状态，禁止自动运动指令下发。

```
bool isJogMode = apiInstance.E_get_Robot_JogStatus ();
```

6.2.189 设置 PSO 控制字使能与禁能指令接口

功能介绍

启用 / 禁用指定轴的 PSO（位置同步输出）功能，PSO 用于轴运动到指定位置时触发同步输出。

参数说明

- AxleID：轴号，数据类型为 int。
- EnableSts：使能状态，数据类型为 bool（true = 使能，false = 禁能）。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

启用 1 轴的 PSO 功能，可配置比较点触发同步输出。

```
bool isSuccess = apiInstance.E_Set_LatchEnable (1, true);
```

6.2.190 获取 PSO 状态字的值接口

功能介绍

查询指定轴的 PSO 功能当前状态（如是否触发、是否报错等）。

参数说明

- AxleID：轴号，数据类型为 int。

返回值

返回 PSO 状态值，数据类型为 int（具体值需参考机器人 SDK 文档定义）。

配置示例

psoStatus 为 1 表示 PSO 已触发，为 0 表示未触发，为负数表示报错。

```
int psoStatus = apiInstance.E_Get_PSOStatus (1);
```

6.2.191 设置指定轴的指定 PSO 比较点指令接口

功能介绍

配置指定轴的 PSO 比较点，当轴运动到该关节位置时触发 PSO 输出，支持最多 16 个关节位置（覆盖多轴联动场景）。

参数说明

- AxleID：轴号，数据类型为 int。
- number：比较点编号，数据类型为 int。
- Apos：关节位置数组，数据类型为 double [16]。

返回值

设置成功返回 true，失败返回 false，数据类型为 bool。

配置示例

为 1 轴配置 1 号比较点，当 1 轴运动到 90° 时触发 PSO 输出。

```
double apos [16] = {90.0, 0.0, 0.0, 0.0, 0.0, 0.0};
bool isSuccess = apiInstance.E_Set_LatchPso (1, 1, apos);
```

6.2.192 执行 MovCircle 指令接口

功能介绍

通过变量名指定辅助点、目标点、速度等参数，执行圆弧插补运动，参数均为已定义的变量名（如示教器中配置的变量）。

参数说明

- PAuxname：辅助位置点变量名，数据类型为 ESString。
- PDstname：目标位置点变量名，数据类型为 ESString。
- velName：速度变量名（默认空），数据类型为 ESString。

- blendType：过渡类型（""、"Rel"、"Abs"三选一，默认空），数据类型为 ESString。
- zname：过渡参数变量名（默认空），数据类型为 ESString。
- tname：工具变量名（默认空），数据类型为 ESString。
- coorname：坐标系变量名（默认空），数据类型为 ESString。
- payloadName：负载变量名（默认空），数据类型为 ESString。
- overrideType：速率类型（""、"GOFF"、"GON" 三选一，默认 "GON"），数据类型为 ESString。
- addDo：附加指令（默认空），数据类型为 const E_ROB_APIBUFF&。
- valfine：到位精度值（默认 0.0），数据类型为 double。
- IsWaitFinish：是否等待完成（默认 false），数据类型为 bool。

返回值

返回指令 JobId，数据类型为 E_ROB_JOBID；id<0 表示失败。

配置示例

通过变量名执行圆弧插补运动，从当前点经 p_aux 到 p_dst，使用 speed100 速度、tool1 工具。

```
E_ROB_JOBID jobId = apiInstance.E_MovCircle ("t_g.p_aux", "t_g.p_dst",
"t_g.speed100", "","t_g.zone0","t_g.tool1", "t_g.user0", "t_g.payload0", "GON");
```

6.2.193 执行 MovCircle2 指令接口

功能介绍

简化版圆弧插补运动指令，直接传入 E_ROB_POS 结构体的辅助点、目标点，无需依赖预定义变量，参数更灵活。

参数说明

- auxP：辅助位置点，数据类型为 const E_ROB_POS&。
- dstPoint：目标位置点，数据类型为 const E_ROB_POS&。
- velo：运行速度（默认值），数据类型为 const E_ROB_SPEED&。
- zone：过渡参数（默认值），数据类型为 const E_ROB_ZONE&。
- payloadId：负载号（默认 0），数据类型为 int。
- toolId：工具号（默认 0），数据类型为 int。
- coordId：用户坐标号（默认 0），数据类型为 int。

- overrideType：速率类型（0=GOVROFF，1=GOVRON，默认 1），数据类型为 int。
- addDo：附加指令（默认空），数据类型为 const E_ROB_APIBUFF&。
- valfine：到位精度值（默认 0.0），数据类型为 double。
- IsWaitFinish：是否等待完成（默认 false），数据类型为 bool。

返回值

返回指令 JobId，数据类型为 E_ROB_JOBID；id<0 表示失败。

配置示例

执行圆弧插补运动，速度 50mm/s，使用默认过渡参数和工具。

```
E_ROB_POS auxP, dstP;
E_ROB_SPEED speed;
 speed.linear = 50;
E_ROB_JOBID jobId = apiInstance.E_MovCircle2 (auxP, dstP, speed);
```

6.2.194 判断当前运动指令是否全部完成接口

功能介绍

检查机器人所有运动队列中的指令是否均已执行完成，是运动结束的全局判断方法。

参数说明

无

返回值

所有运动完成返回 true，仍有未完成运动返回 false，数据类型为 bool。

配置示例

循环等待直到所有运动指令执行完成。

```
while (!apiInstance.E_Waitcomplete ())
{
// 等待运动完成，可添加延时避免高频查询
Sleep (100);
}
```

6.2.195 机器人停止清空运动队列接口

功能介绍

停止机器人当前运动，并清空未执行的运动队列，终止所有待执行的运动指令。

参数说明

无

返回值

操作成功返回 true，失败返回 false，数据类型为 bool。

配置示例

清空运动队列，终止所有未执行的运动指令；通常在紧急停止场景使用。

```
bool isSuccess = apiInstance.E_ClearBuffer ();
```

6.2.196 停止机器人并清空队列接口

功能介绍

立即停止机器人当前运动，清空所有未执行的运动队列，使机器人进入 Stop 模式，禁止新运动指令执行（需 E_MotionStart_DC 唤醒）。

参数说明

无

返回值

操作成功返回 true，失败返回 false，数据类型为 bool

配置示例

紧急停止机器人运动并清空队列，进入 Stop 模式，适用于异常场景。

```
bool isSuccess = apiInstance.E_MotionStop_DC ();
```

6.2.197 SCARA 机型 3 轴抱闸控制接口

功能介绍

专门用于 SCARA 机型的 3 轴抱闸状态控制，支持解锁 / 锁定抱闸操作，仅对 SCARA 机型生效。

参数说明

- status：抱闸状态，数据类型为 bool：true = 解抱闸（释放抱闸，轴可运动）、false = 锁抱闸（抱紧抱闸，轴固定）。

返回值

操作成功返回 true，失败返回 false，数据类型为 bool。

配置示例

仅适用于 SCARA 机型 3 轴，解抱闸后轴可正常运动，锁抱闸后轴被机械锁定，禁止运动（紧急固定场景使用）。

```
bool isSuccess = apiInstance.E_Release_brake_scara (true);
bool isSuccess = apiInstance.E_Release_brake_scara (false);
```

6.2.198 获取抱闸状态接口

功能介绍

查询 SCARA 机型 3 轴当前的抱闸状态，仅对 SCARA 机型生效。

参数说明

无

返回值

返回状态码，数据类型为 int：0 = 获取失败、1 = 解抱闸状态、2 = 锁抱闸。

配置示例

通过状态码判断抱闸当前状态，避免在锁抱闸时下发运动指令导致报错。

```
int brakeStatus = apiInstance.E_Get_brake_scara ();
if (brakeStatus == 1) {
}
else if (brakeStatus == 2) {
}
```

6.2.199 笛卡尔坐标转换（坐标系映射）接口

功能介绍

该接口并非简单的笛卡尔坐标与关节坐标转换，而是实现不同坐标系下笛卡尔坐标的映射转换：将源坐标系（源工具 + 源用户坐标）下的笛卡尔位置，转换为目标坐标系（目标工具 + 目标用户坐标）下的笛卡尔位置。

参数说明

- dstPos：输出参数，转换后的目标笛卡尔位置，数据类型为 E_ROB_POS&。
- srcPos：输入参数，源笛卡尔位置（待转换的原始位置），数据类型为 E_ROB_POS

- srcPos：输入参数，源笛卡尔位置（待转换的原始位置），数据类型为 E_ROB_POS。
- dstToolCoord：输入参数，目标工具坐标系，数据类型为 E_ROB_COORD。
- dstusrCoord：输入参数，目标用户坐标系，数据类型为 E_ROB_COORD。
- srcToolCoord：输入参数，源工具坐标系，数据类型为 E_ROB_COORD。
- srcusrCoord：输入参数，源用户坐标系，数据类型为 E_ROB_COORD。

返回值

转换成功返回 true，失败返回 false，数据类型为 bool。

配置示例

常用于更换工具 / 切换用户坐标系后，将原位置映射到新坐标系下，避免重新示教点位；例如更换焊枪（工具坐标系变化）后，将原焊接点位转换为新工具坐标系下的位置。

```
E_ROB_COORD srcTool, srcUser, dstTool, dstUser;
E_ROB_POS srcPos;
srcPos.x = 100;
srcPos.y = 50;
srcPos.z = 20;
E_ROB_POS dstPos;
bool isSuccess = apiInstance.E_CposToCPos (dstPos, srcPos, dstTool, dstUser, srcTool, srcUser);
```

错误列表

错误号	错误信息	处理方法
88801	APIServer 断开连接	重新启动APIServer程序
88802	执行运动指令中,不允许设置JOG使能	先停止当前运动，在设置JOG使能
88803	JOG使能中,不可以执行运动指令	先取消JOG使能，然后调用运动接口
88804	执行运动指令中,不允许设置Mot使能	先停止当前运动，在设置Mot使能
88805	API组件初始化失败	重新启动控制器
88806	当前TOOL加载中,不允许修改	先卸载当前TOOL（即加载nulltool（ID为0）），再进行修改
88807	当前UserCoord加载中,不允许修改	先卸载当前UserCoord（即加载world（ID为0）），再进行修改
88808	当前PayLoad加载中,不允许修改	先卸载当前PayLoad（即加载Full PayLoad（ID为0）），再进行修改
88809	当前TOOL不存在,修改失败	先创建对应ID的TOOL变量，再进行修改
88810	当前UserCoord不存在,修改失败	先创建对应ID的UserCoord变量，再进行修改
88811	当前PayLoad不存在,修改失败	先创建对应ID的PayLoad变量，再进行修改
88812	当前TOOL不存在,加载失败	选择加载存在的TOOL
88813	当前UserCoord不存在,加载失败	选择加载存在的UserCoord
88814	当前PayLoad不存在,加载失败	选择加载存在的PayLoad
88815	机器人运动中,不允许加载	先停止当前运动，再加载Tool或UserCoord或PayLoad
88816	当前机器人不在暂停状态,继续运动失败	重新发送运动指令