

# PythonSDK 使用手册

JK-RDQK-UM-021

Python 二次开发接口文档

## 版本记录

版本编号	版本日期	支持控制器的版本	说明
V1.0.0	2020.3.24	V1.4.10/V2.0.10 及以上	1. 创建文档
V1.0.0	2020.6.24	V1.4.10/V2.0.10 及以上	1. 增加对 joint_move、linear_move、servo_j 等不同运动指令的补充说明
V1.0.14	2020.7.24	V1.4.10/V2.0.10 及以上	1. 新增接口 4.61 到 4.64 2. 对 jog_stop 定义进行修改
V1.0.15	2021.04.27	V1.4.24/V1.5.12.1 7/V2.0.24 及以上	新增接口 4.69 到 4.110
V1.0.18	2021.08.30	V1.4.24/V1.5.12.1 7/V2.0.24 及以上	增加 API 使用说明
V2.0.1	2021.12.10	V1.4.24/V1.5.12.1 7/V2.0.24 及以上	增加 ftp 接口

## 目录

1. 简介.....	- 6 -
2. 文档须知.....	- 6 -
2.1 linux 下的使用.....	- 6 -
2.2 windows 下的使用.....	- 6 -
2.3 动态库版本号查询方法.....	- 6 -
3. 数据类型.....	- 7 -
3.1 IO 类型.....	- 7 -
3.2 坐标系类型.....	- 7 -
3.3 运动类型.....	- 7 -
3.4 函数返回值.....	- 7 -
4. 接口.....	- 8 -
4.1 实例化机器人.....	- 8 -
4.2 5 机器人登录.....	- 8 -
4.3 机器人注销.....	- 8 -
4.4 打开机器人电源.....	- 9 -
4.5 关闭机器人电源.....	- 9 -
4.6 机器人控制柜关机.....	- 9 -
4.7 控制机器人上使能.....	- 10 -
4.8 控制机器人下使能.....	- 10 -
4.9 查询 SDK 版本号.....	- 10 -
4.10 获取控制器 IP.....	- 11 -
4.11 控制机器人手动模式下运动.....	- 11 -
4.12 控制机器人手动模式下运动停止.....	- 13 -
4.13 获取当前机器人的六个关节角度值.....	- 13 -
4.14 获取当前设置下工具末端的位姿.....	- 14 -
4.15 求机器人逆解.....	- 15 -
4.16 求机器人正解.....	- 15 -
4.17 机器人关节运动.....	- 16 -
4.18 机器人末端直线运动.....	- 16 -
4.19 机器人 SERVO MOVE 模式使能.....	- 17 -
4.20 机器人关节空间伺服模式运动.....	- 17 -
4.21 机器人关节空间伺服模式运动扩展.....	- 19 -
4.22 机器人笛卡尔空间伺服模式运动.....	- 19 -
4.23 机器人笛卡尔空间伺服模式运动扩展.....	- 21 -
4.24 设置数字输出变量(DO)的值.....	- 21 -
4.25 设置模拟输出变量的值(AO)的值.....	- 22 -
4.26 查询数字输入(DI)状态.....	- 23 -
4.27 查询数字输出(DO)状态.....	- 23 -
4.28 查询模拟量输入变量(AI)的值.....	- 23 -

4.29 查询模拟量输出变量(AO)的值.....	23 -
4.30 查询扩展 IO 模块是否运行.....	24 -
4.31 加载指定的作业程序.....	24 -
4.32 获取已加载的作业程序名称.....	24 -
4.33 获取当前机器人作业程序的执行行号.....	25 -
4.34 运行当前加载的作业程序.....	25 -
4.35 暂停当前运行的作业程序.....	25 -
4.36 继续运行当前暂停的作业程序.....	25 -
4.37 终止当前执行的作业程序.....	26 -
4.38 获取机器人作业程序执行状态.....	26 -
4.39 设置机器人运行速度倍率.....	27 -
4.40 获取机器人运行速度倍率.....	27 -
4.41 设置指定编号的工具信息.....	27 -
4.42 设置当前使用的工具 ID.....	27 -
4.43 查询目标工具坐标系的信息.....	28 -
4.44 查询机器人当前使用的工具 ID.....	28 -
4.45 设置用户坐标系信息.....	29 -
4.46 获取用户坐标系信息.....	29 -
4.47 设置当前使用的用户坐标系 ID.....	29 -
4.48 查询当前使用的用户坐标系 ID.....	30 -
4.49 控制器机器人进入或退出拖拽模式.....	30 -
4.50 查询机器人是否处于拖拽模式.....	31 -
4.51 获取机器人状态.....	31 -
4.52 查询机器人是否超出限位.....	31 -
4.53 查询机器人运动是否停止.....	32 -
4.54 查询机器人是否处于碰撞保护模式.....	32 -
4.55 碰撞之后从碰撞保护模式恢复.....	32 -
4.56 设置机器人碰撞等级.....	33 -
4.57 获取机器人碰撞等级.....	33 -
4.58 欧拉角到旋转矩阵的转换.....	34 -
4.59 旋转矩阵到欧拉角的转换.....	34 -
4.60 四元数到旋转矩阵的转换.....	34 -
4.61 旋转矩阵到四元数的转换.....	34 -
4.62 注册机器人出现错误时的回调函数.....	35 -
4.63 机器人负载设置.....	35 -
4.64 获取机器人负载数据.....	35 -
4.65 终止当前机械臂运动.....	36 -
4.66 获取机器人状态监测数据.....	37 -
4.67 设置 SDK 是否开启调试模式.....	38 -
4.68 设置错误码文件的路径.....	38 -
4.69 获取最新的错误码.....	38 -
4.70 错误状态清除.....	39 -
4.71 设置轨迹复现配置参数.....	39 -
4.72 获取轨迹复现配置参数.....	40 -

4.73 采集轨迹复现数据控制开关.....	- 41 -
4.74 采集轨迹复现数据状态查询.....	- 41 -
4.75 查询控制器中已经存在的轨迹复现数据的文件名.....	- 41 -
4.76 重命名轨迹复现数据的文件名.....	- 42 -
4.77 删除控制器中轨迹复现数据的文件.....	- 42 -
4.78 控制器中轨迹复现数据文件生成控制器执行脚本.....	- 42 -
4.79 机器人扩展关节运动.....	- 43 -
4.80 机器人扩展末端直线运动.....	- 43 -
4.81 机器人末端圆弧运动.....	- 43 -
4.82 机器人 SERVO 模式下禁用滤波器.....	- 44 -
4.83 机器人 SERVO 模式下关节空间一阶低通滤波.....	- 44 -
4.84 机器人 SERVO 模式下关节空间非线性滤波.....	- 45 -
4.85 机器人 SERVO 模式下笛卡尔空间非线性滤波.....	- 45 -
4.86 机器人 SERVO 模式下关节空间多阶均值滤波.....	- 46 -
4.87 SERVO 模式下速度前瞻参数设置.....	- 46 -
4.88 设置 SDK 日志路径.....	- 46 -
4.89 设置传感器品牌.....	- 47 -
4.90 获取传感器品牌.....	- 47 -
4.91 开启或关闭力矩传感器.....	- 48 -
4.92 设置柔顺控制参数.....	- 48 -
4.93 开始辨识工具末端负载.....	- 48 -
4.94 获取末端负载辨识状态.....	- 50 -
4.95 获取末端负载辨识结果.....	- 50 -
4.96 设置传感器末端负载.....	- 50 -
4.97 获取传感器末端负载.....	- 50 -
4.98 力控导纳使能控制.....	- 51 -
4.99 设置力控类型和传感器初始化状态.....	- 52 -
4.100 获取力控类型和传感器初始化状态.....	- 52 -
4.101 获取力控柔顺控制参数.....	- 52 -
4.102 设置力矩传感器 IP 地址.....	- 53 -
4.103 获取力矩传感器 IP 地址.....	- 53 -
4.104 关闭力矩控制.....	- 53 -
4.105 设置速度柔顺控制参数.....	- 54 -
4.106 设置柔顺控制力矩条件.....	- 54 -
4.107 设置网络异常时机器人自动终止运动类型.....	- 54 -
4.108 设置机器人状态数据自动更新时间间隔.....	- 55 -
4.109 查询控制器 ftp 的目录.....	- 56 -
4.110 下载 FTP 文件.....	- 56 -
4.111 上传文件到 ftp.....	- 57 -
4.112 ftp 初始化.....	- 57 -
4.113 ftp 关闭.....	- 57 -
4.114 重命名 ftp 上的文件.....	- 58 -
4.115 删除文件到 ftp.....	- 58 -
5. 反馈和勘误.....	- 59 -

# 1.简介

本文档为 python 版本的二次开发接口文档。

## 2.文档须知

- 运行环境：linux python3.5 32 位 、 windows Python3.7 64 位
- 使用到参数的单位：毫米(mm)，弧度(rad)
- 非特别说明的代码示例中都默认机器人已经开机，并且上电上使能
- 文档中的所有代码示例都默认在机器人的工作空间内没有任何干涉
- 文档中的示例都已经默认用户的 Python 解释器能够找到 jkrc 模块

### 2.1 linux 下的使用

Linux 需要将 libjakaAPI.so 和 jkrc.so 放在同一个文件夹下，并添加当前文件夹路径到环境变量，  
`export LD_LIBRARY_PATH=/xx/xx/`

### 2.2 windows 下的使用

Windows 需要将 jkrc 以及 jakaAPI.dll 放在同一个文件夹下，常见问题可以查询 FAQ。

### 2.3 动态库版本号查询方法

jkrc 的正常使用需要包含动态库文件，以下是动态库版本号的查询方法：

Windows 中右击 dll 文件，选择属性，在“详细信息”选项卡中可以看到版本信息。

Linux 中输入命令 `strings libjakaAPI.so | grep jakaAPI_version` 查询版本信息。

## 3.数据类型

### 3.1 IO 类型

JAKA 机器人有三种 IO ， 分别是控制柜 IO，工具 IO，扩展 IO。

以下定义在后面的示例代码中会使用到：

```
IO_CABINET  =0  #控制柜面板 IO
IO_TOOL    = 1    #工具 IO
IO_EXTEND   = 2    #扩展 IO
```

### 3.2 坐标系类型

JAKA 机器人有三种坐标系，分别是基坐标系/当前的用户坐标系，关节空间，工具坐标系

以下定义在后面的示例代码中会使用到：

```
COORD_BASE = 0    # 世界或者当前的用户坐标系
COORD_JOINT = 1    # 关节空间
COORD_TOOL  = 2    #工具坐标系
```

### 3.3 运动类型

JAKA 机器人运动类型有两种，定义如下：

```
ABS = 0    # 绝对运动
INCR = 1    # 增量运动
```

### 3.4 函数返回值

类型	返回值	描述
ERR_SUCC	0	成功
ERR_INVALID_HANDLER	-1	无效的句柄
ERR_INVALID_PARAMETER	-2	无效的参数
ERR_COMMUNICATION_ERR	-3	通信错误
ERR_KINE_INVERSE_ERR	-4	逆解失败
ERR_EMERGENCY_PRESSED	-5	急停键没有松开
ERR_NOT_POWERED	-6	机器人未上电
ERR_NOT_ENABLED	-7	机器人未使能
ERR_DISABLE_SERVOMODE	-8	未进入 SERVOMODE 模式

ERR_NOT_OFF_ENABLE	-9	关闭电源前未先下使能
ERR_PROGRAM_IS_RUNNING	-10	程序正在运行
ERR_CANNOT_OPEN_FILE	-11	打开文件失败

除了实例化机器人对象之外的其它接口的返回值都是一个元组。Get 类的接口的返回值形式为 **(errcode,data)**，第一个元素是 errcode 错误码，成功调用 errcode 为 0，如果发生错误可能得到的错误码值可以查询上表。第二个元素是 data，如关节角度值等等。

例如使用 **get\_joint\_position()** 获取关节角度时：

成功获取关节角度: (0,[1,2,3,4,5,6])

因为通信错误，无法获得关节角度时: (-3,)

## 4. 接口

### 4.1 实例化机器人

函数	RC(ip)
描述	实例化一个机器人对象
参数	ip: 机器人的 IP 地址，需要填入一个字符串只有正确的 IP 地址实例化出的对象才能控制机器人。
返回值	成功：返回一个机器人对象 失败：创建的对象会被销毁

代码示例：

```
import jkrc
robot = jkrc.RC("192.168.2.64") #返回一个机器人对象
```

### 4.2 5 机器人登录

函数	login()
描述	连接机器人控制器
参数	
返回值	成功: (0,) 失败: 其它

### 4.3 机器人注销

函数	logout()
描述	断开控制器连接
参数	

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu



返回值	成功: (0,) 失败: 其它
-----	--------------------

代码示例:

```
import jkrc
robot = jkrc.RC("192.168.2.64") # 返回一个机器人对象
robot.login() # 登录
pass
robot.logout() # 登出
```

## 4.4 打开机器人电源

函数	power_on()
描述	打开机器人电源, 给真实的机器人上电大概会有 8 秒钟左右的延迟
参数	
返回值	成功: (0,) 失败: 其它

代码示例:

```
import jkrc
robot = jkrc.RC("192.168.2.64") # 返回一个机器人对象
robot.login() # 登录
robot.power_on() # 上电
robot.logout() # 登出
```

## 4.5 关闭机器人电源

函数	power_off()
描述	关闭机器人电源
参数	
返回值	成功: (0,) 失败: 其它

## 4.6 机器人控制柜关机

函数	shut_down()
描述	机器人控制柜关机
参数	
返回值	成功: (0,) 失败: 其它

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web: www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

## 4.7 控制机器人上使能

函数	<b>enable_robot()</b>
描述	控制机器人上使能
参数	
返回值	成功: (0,) 失败: 其它

代码示例:

```
import jkrc
robot = jkrc.RC("192.168.2.64") # 返回一个机器人对象
robot.login() # 登录
robot.enable_robot() #
robot.logout() # 登出
```

## 4.8 控制机器人下使能

函数	<b>disable_robot()</b>
描述	控制机器人下使能
参数	
返回值	成功: (0,) 失败: 其它

## 4.9 查询 SDK 版本号

函数	<b>get_sdk_version()</b>
描述	获取 SDK 版本号
参数	
返回值	成功: (0,version) 失败: 其它

代码示例:

```
''' 查询 sdk 版本号 '''
import jkrc
robot = jkrc.RC("192.168.2.64") #
robot.login() #
ret = robot.get_sdk_version()
print("SDK version is:", ret[1])
robot.logout() # 登出
```

## 4.10 获取控制器 IP

函数	<code>get_controller_ip()</code>
描述	获取控制器 IP
参数	
返回值	成功: (0, ip_list), ip_list: 控制器 ip 列表, 控制器名字为具体值时返回该名字所对应的控制器 IP 地址, 控制器名字为空时, 返回网段类内的所有控制器 IP 地址 失败: 其它

## 4.11 控制机器人手动模式下运动

函数	<code>jog(aj_num, move_mode, coord_type, jog_vel, pos_cmd)</code>
描述	控制机器人手动模式下运动
参数	aj_num: axis_joint_based 标识值, 在关节空间下代表轴号, 1 轴到六轴的轴号分别对应数字 0 到 5, 笛卡尔空间下依次为 x, y, z, rx, ry, rz 分别对应数字 0 到 5 move_mode: 0 代表绝对运动, 1 代表增量运动, 2 代表连续运动 coord_type: 机器人运动坐标系, 工具坐标系, 基坐标系 (当前的世界/用户坐标系) 或关节空间 jog_vel: 指令速度, 旋转轴或关节运动单位为 rad/s, 移动轴单位为 mm/s, 速度的正负决定运动方向的正负。 pos_cmd: 指令位置, 旋转轴或关节运动单位为 rad, 移动轴单位为 mm, 当 move_mdoe 是绝对运动时参数可忽略
返回值	成功: (0,) 失败: 其它

代码示例:

```
'''jog 运动'''
'''1.关节空间 jog'''

1.  #-*- coding: utf-8 -*-
2.  import sys
3.  sys.path.append('D:\\vs2019ws\\PythonCtt\\PythonCtt')
4.  import time
5.  import jkrc
6.  PI=3.1415926
7.  #坐标系
8.  COORD_BASE = 0
9.  COORD_JOINT = 1
10. COORD_TOOL = 2
11. #运动模式
12. ABS = 0
13. INCR= 1
```

```
14. #关节 1~6 依次对应 0~5,
15.
16. robot = jkrc.RC("192.168.2.194")#返回机器人对象
17. robot.login()#登录
18. robot.power_on() #上电
19. robot.enable_robot()
20. print("move1")
21. robot.jog(0,INCR,COORD_JOINT,30*PI/180,PI/2)
22. time.sleep(5)#jog 为非阻塞指令, 运动状态下接收 jog 指令会被丢弃
23. print("move2")
24. robot.jog(0,INCR,COORD_JOINT,5,-PI/2)
25. time.sleep(3)
26. robot.jog_stop()
27. robot.logout()
```

## '''2.笛卡尔空间 jog'''

```
1. import jkrc
2. import time
3. COORD_BASE = 0    # 基坐标系
4. COORD_JOINT = 1   # 关节空间
5. COORD_TOOL = 2    #工具坐标系
6. ABS = 0           # 绝对运动
7. INCR = 1          # 增量运动
8. cart_x = 0        #x 方向
9. cart_y = 1        #y 方向
10. cart_z = 2       #z 方向
11. cart_rx = 3       #rx 方向
12. cart_ry = 4       #ry 方向
13. cart_rz = 5       #rz 方向
14. robot = jkrc.RC("192.168.2.64")#返回一个机器人对象
15. robot.login()     #登录
16. robot.jog(aj_num = cart_z ,move_mode = INCR,coord_type = COORD_BASE ,jog_vel = 5,pos_cmd =
    10)    # z 正方向运动 10mm
17. robot.jog_stop()
18. robot.logout()    #登出
```

说明:

若机器人已经接近奇异姿态或达到关节限位, 使用上述示例代码, 将无法 jog 机器人

## 4.12 控制机器人手动模式下运动停止

函数	<code>jog_stop(joint_num)</code>
描述	控制机器人手动模式下运动停止，用于停止 jog
参数	joint_num: num 代表要停止运动的关节轴号，轴号 0 到 5，分别代表关节 1 到关节 6。值的注意的是-1 代表停止所有轴的运动。
返回值	成功：(0,) 失败：其它

代码示例：

```

1.  #-*- coding: utf-8 -*-
2.  import sys
3.  sys.path.append('D:\\vs2019ws\\PythonCtt\\PythonCtt')
4.  import time
5.  import jkrc
6.  PI=3.1415926
7.  #坐标系
8.  COORD_BASE = 0
9.  COORD_JOINT = 1
10. COORD_TOOL = 2
11. #运动模式
12. ABS = 0
13. INCR= 1
14. #关节 1~6 依次对应 0~5,
15.
16. robot = jkrc.RC("192.168.2.160")#返回机器人对象
17. robot.login()#登录
18. robot.power_on() #上电
19. robot.enable_robot()
20. print("move1")
21. robot.jog(0,INCR,COORD_JOINT,30*PI/180,PI/2)
22. time.sleep(5)#jog 为非阻塞指令，运动状态下接收 jog 指令会被丢弃
23. print("move2")
24. robot.jog(0,INCR,COORD_JOINT,5,-PI/2)
25. time.sleep(0.5)
26. robot.jog_stop(0) #运动 0.5 秒后停止，与前一个示例代码对比
27. robot.logout()

```

## 4.13 获取当前机器人的六个关节角度值

函数	<code>get_joint_position()</code>
描述	获取当前机器人的六个关节角度值

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海：上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州：江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

参数	
返回值	成功: (0, joint_pos), joint_pos 是一个包含 6 位元素的元组 (j1, j2, j3, j4, j5, j6), j1, j2, j3, j4, j5, j6 分别代表关节 1 到关节 6 的角度值。 失败: 其它

代码示例:

```
'''获得关节角度'''
1. # -*- coding: utf-8 -*-
2. import sys
3. #sys.path.append('D:\\vs2019ws\\PythonCtt\\lib')
4. import time
5. import jkrc
6. PI=3.1415926
7. robot = jkrc.RC("192.168.2.160")#返回一个机器人对象
8. ret = robot.login()#登录
9. ret = robot.get_joint_position()
10. if ret[0] == 0:
11.     print("the joint position is :",ret[1])
12. else:
13.     print("some things happend,the errcode is: ",ret[0])
14. robot.logout() #登出
```

## 4.14 获取当前设置下工具末端的位姿

函数	get_tcp_position()
描述	获取当前设置下工具末端的位姿
参数	
返回值	成功: (0, cartesian_pose), cartesian_pose 是一个包含 6 位元素的元组(x, y, z, rx, ry, rz), x, y, z, rx, ry, rz 代表机器人工具末端的位姿值 失败: 其它

代码示例:

```
'''获得 tcp 位姿'''
1. # -*- coding: utf-8 -*-
2. import sys
3. #sys.path.append('D:\\vs2019ws\\PythonCtt\\lib')
4. import time
5. import jkrc
6. PI=3.1415926
7. robot = jkrc.RC("192.168.2.160")#返回一个机器人对象
8. ret = robot.login()#登录
```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```

9. ret = robot.get_tcp_position()
10. if ret[0] == 0:
11.     print("the tcp position is :",ret[1])
12. else:
13.     print("some things happend,the errcode is: ",ret[0])
14. robot.logout() #登出

```

## 4.15 求机器人逆解

函数	kine_inverse(ref_pos, cartesian_pose)
描述	计算指定位姿在当前工具、当前安装角度以及当前用户坐标系设置下的逆解。
参数	ref_pos: 关节空间参考位置, 建议选用机器人当前关节位置。 cartesian_pose: 笛卡尔空间位姿计算结果。
返回值	成功: (0, joint_pos) joint_pos 是一个包含 6 位元素的元组 (j1, j2, j3, j4, j5, j6), j1, j2, j3, j4, j5, j6 分别代表关节 1 到关节 6 的角度值 失败: 其它

## 4.16 求机器人正解

函数	kine_forward(joint_pos)
描述	计算指定关节位置在当前工具、当前安装角度以及当前用户坐标系设置下的位姿值
参数	joint_pos: 关节空间位置
返回值	成功: (0, cartesian_pose), cartesian_pose 是一个包含 6 位元素的元组(x, y, z, rx, ry, rz), x, y, z, rx, ry, rz 代表机器人工具末端的位姿值 失败: 其它

代码示例:

```

'''机器人正解'''
1. import jkrc
2. robot = jkrc.RC("192.168.2.64") #返回一个机器人对象
3. robot.login() #登录
4. ret = robot.get_joint_position()
5. if ret[0] == 0:
6.     print("the joint position is :",ret[1])
7. else:
8.     print("some things happend,the errcode is: ",ret[0])
9. joint_pos = ret[1]
10. robot.kine_forward(joint_pos) #求机器人正解
11. robot.logout() #登出

```

## 4.17 机器人关节运动

函数	<code>joint_move(joint_pos, move_mode, is_block, speed)</code>
描述	机器人关节运动到目标点位
参数	<code>joint_pos</code> : 机器人关节运动目标位置。 <code>move_mode</code> : 0 代表绝对运动, 1 代表相对运动 <code>is_block</code> : 设置接口是否为阻塞接口, <code>TRUE</code> 为阻塞接口 <code>FALSE</code> 为非阻塞接口, 阻塞表示机器人运动完成才会有返回值, 非阻塞表示接口调用完成立刻就有返回值。 <code>speed</code> : 机器人关节运动速度, 单位: rad/s
返回值	成功: (0,) 失败: 其它

代码示例:

```
'''关节运动'''
1.  # -*- coding: utf-8 -*-
2.  # import sys
3.  # sys.path.append('D:\\vs2019ws\\PythonCtt\\PythonCtt')
4.  import time
5.  import jkrc
6.  PI=3.1415926
7.  #运动模式
8.  ABS = 0
9.  INCR= 1
10. joint_pos=[PI/2,PI/3,0,PI/4,0,0]
11. robot = jkrc.RC("192.168.2.160")#返回机器人对象
12. robot.login()#登录
13. robot.power_on() #上电
14. robot.enable_robot()
15. print("move1")
16. robot.joint_move(joint_pos,ABS,True,1)
17. time.sleep(3)
18. robot.logout()
```

## 4.18 机器人末端直线运动

函数	<code>linear_move(end_pos, move_mode, is_block, speed)</code>
描述	机器人末端直线运动到目标点位
参数	<code>end_pos</code> : 机器人末端运动目标位置 <code>move_mode</code> : 0 代表绝对运动, 1 代表相对运动 <code>is_block</code> : 设置接口是否为阻塞接口, <code>TRUE</code> 为阻塞接口 <code>FALSE</code> 为非阻塞接口, 阻塞表示机器人运动完成才会有返回值, 非阻塞表示接口调用完成立刻

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu



	就有返回值。 speed: 机器人直线运动速度，单位：mm/s
返回值	成功：(0,) 失败：其它

代码示例：

'''直线运动'''

```

1. import time
2. import jkrc
3. PI=3.1415926
4. #运动模式
5. ABS = 0
6. INCR= 1
7. tcp_pos=[0,0,-30,0,0,0]
8. robot = jkrc.RC("192.168.2.160")#返回机器人对象
9. robot.login()#登录
10. robot.power_on() #上电
11. robot.enable_robot()
12. print("move1")
13. #阻塞 沿 z 轴负方向 以 10mm/s 运动 30mm
14. ret=robot.linear_move(tcp_pos,INCR,True,10)
15. print(ret[0])
16. time.sleep(3)
17. robot.logout()

```

说明:由于机器人型号的不同,上述示例填写的位姿可能会逆解失败,返回 -4.

## 4.19 机器人 SERVO MOVE 模式使能

函数	<b>servo_move_enable(enable)</b>
描述	机器人 SERVO MOVE 模式使能
参数	enable : TRUE 为进入 SERVO MOVE 模式, FALSE 表示退出该模式
返回值	成功：(0,) 失败：其它

## 4.20 机器人关节空间伺服模式运动

函数	<b>servo_j(joint_pos, move_mode)</b>
描述	机器人关节空间位置控制模式,需要注意的事项如下: (a) 当用户使用这个接口的时候需要先调用 servo_move_enable(True), 进入位置控制模式 (b) 这条指令一般是高校科研当中做轨迹规划时使用。

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

	<p>(c) 当用户使用此模式控制机器人的运动的时候，控制器的规划器不参与运动的插补，位置指令会直接发给伺服，所以用户需要自己进行轨迹规划，否则机器人的运动效果会比较差，如抖动剧烈等现象，无法达到用户的预期。</p> <p>(d) 由于控制器的控制周期为 8ms，建议用户的发送周期也为 8ms，并且需要连续发送，只发一次没有效果。如果网络状况较差，发送周期可以小于 8ms。</p> <p>(e) Jaka 机器人的关节速度上限是 180 度/秒，如果发送的关节角度使关节速度超过了此上限。这条指令就会失效。比如发送的关节角度是[1.5,0.5,0.5,0.5,0.5,0.5](此处单位是度)，发送周期是 8ms，那么 <math>1.5/0.008 = 187.5</math> 度/秒，超过了关节速度上限。那么指令将会失效。</p> <p>(f) 使用完这条指令，需要使用 servo_move_enable(False),退出位置控制模式。</p> <p>(g) 这条指令和前文提到的 joint_move()区别较大，joint_move 的插补是由控制器进行的，用户无需关心。用户使用 servo_j 指令时需要预先进行轨迹规划，否则使用效果会很差，无法达到预期。<b>无特殊需要，机器人在关节空间的运动建议使用 joint_move，而不是 servo_j。</b></p>
参数	<p>joint_pos: 机器人关节运动目标位置。</p> <p>move_mode 指定运动模式：0 为绝对运动，1 为增量运动</p>
返回值	<p>成功：(0,)</p> <p>失败：其它</p>

代码示例：

'''servo\_j 的使用'''

```

1.  # -*- coding: utf-8 -*-
2.  import sys
3.  sys.path.append('D:\\vs2019ws\\PythonCtt\\PythonCtt')
4.  import time
5.  import jkrc
6.  PI=3.1415926
7.
8.  ABS = 0          # 绝对运动
9.  INCR = 1         # 增量运动
10. Enable = True
11. Disable = False
12.
13. robot = jkrc.RC("192.168.2.160")#返回一个机器人对象
14. robot.login()#登录
15. robot.power_on() #上电
16. robot.enable_robot()
17. robot.servo_move_enable(Enable) #进入位置控制模式
18. print("enable")
19. time.sleep(0.5)
20. for i in range(200):
21.     robot.servo_j(joint_pos =[0.001,0,0,0,0,0.001],move_mode = INCR)#
22.     time.sleep(0.008)
23. for i in range(200):
24.     robot.servo_j(joint_pos =[-0.001,0,0,0,0,-0.001],move_mode = INCR)

```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海：上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州：江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```

25.     time.sleep(0.008)
26. robot.servo_move_enable(Disable)#退出位置控制模式
27. print("disable")
28. robot.logout() #登出

```

## 4.21 机器人关节空间伺服模式运动扩展

函数	<code>servo_j_extend(joint_pos, move_mode, setp_num)</code>
描述	<p>(a) 机器人关节空间位置控制模式,需要注意的事项如下:</p> <p>(b) 当用户使用这个接口的时候需要先调用 <code>servo_move_enable(True)</code>, 进入位置控制模式</p> <p>(c) 这条指令一般是高校科研当中做轨迹规划时使用。</p> <p>(d) 当用户使用此模式控制机器人的运动的时候, 控制器的规划器不参与运动的插补, 位置指令会直接发给伺服, 所以用户需要自己进行轨迹规划, 否则机器人的运动效果会比较差, 如抖动剧烈等现象, 无法达到用户的预期。</p> <p>(e) 由于控制器的控制周期为 8ms, 建议用户的发送周期也为 8ms, 并且需要连续发送, 只发一次没有效果。如果网络状况较差, 发送周期可以小于 8ms。</p> <p>(f) Jaka 机器人的关节速度上限是 180 度/秒, 如果发送的关节角度使关节速度超过了此上限。这条指令就会失效。比如发送的关节角度是[1.5,0.5,0.5,0.5,0.5,0.5](此处单位是度), 发送周期是 8ms, 那么 <math>1.5/0.008 = 187.5</math> 度/秒, 超过了关节速度上限。那么指令将会失效。</p> <p>(g) 使用完这条指令, 需要使用 <code>servo_move_enable(False)</code>,退出位置控制模式。</p> <p>(h) 这条指令和前文提到的 <code>joint_move()</code> 区别较大, <code>joint_move</code> 的插补是由控制器进行的, 用户无需关心。用户使用 <code>servo_j</code> 指令时需要预先进行轨迹规划, 否则使用效果会很差, 无法达到预期。无特殊需要, 机器人在关节空间的运动建议使用 <code>joint_move</code>, 而不是 <code>servo_j</code>。</p>
参数	<p><code>joint_pos</code>: 机器人关节运动目标位置。</p> <p><code>move_mode</code> 指定运动模式: 0 为增量运动, 1 为绝对运动</p> <p><code>step_num</code>: 倍分周期, <code>servo_j</code> 运动周期为 <code>step_num*8ms</code>, 其中 <code>step_num</code> ≥ 1。</p>
返回值	<p>成功: (0,)</p> <p>失败: 其它</p>

## 4.22 机器人笛卡尔空间伺服模式运动

函数	<code>servo_p(cartesian_pose, move_mode)</code>
描述	<p>机器人笛卡尔空间位置控制模式,需要注意的事项如下:</p> <p>(a) 当用户使用这个接口的时候需要先调用 <code>servo_move_enable(True)</code>, 进入位置控制模式</p> <p>(b) 这条指令一般是高校科研当中做轨迹规划时使用。</p> <p>(c) 当用户使用此模式控制机器人的运动的时候, 控制器的规划器不参与运动的插补, 控制器进行逆解后, 会将位置指令会直接发给伺服, 所以用户需要自己进行</p>

	<p>轨迹规划，否则机器人的运动效果会比较差，如抖动剧烈等现象，无法达到用户的预期。</p> <p>(d) 由于控制器的控制周期为 8ms，建议用户的发送周期也为 8ms，并且需要连续发送，只发一次没有效果。如果网络状况较差，发送周期可以小于 8ms。</p> <p>(e) Jaka 机器人的关节速度上限是 180 度/秒，如果发送的位置使得机器人的关节速度超过了此上限。这条指令就会失效。</p> <p>(f) 使用完这条指令，需要使用 <code>servo_move_enable(False)</code>，退出位置控制模式。</p> <p>(g) 这条指令和前文提到的 <code>linear_move()</code> 区别较大，<code>linear_move</code> 的插补是由控制器进行的，用户无需关心。用户使用 <code>servo_p</code> 指令时需要预先进行轨迹的规划，否则使用效果会很差，无法达到预期。<b>无特殊需要，机器人在笛卡尔空间的运动建议使用 <code>linear_move</code>，而不是 <code>servo_p</code>。</b></p>
参数	<p><code>cartesian_pose</code>: 机器人笛卡尔空间运动目标位置。</p> <p><code>move_mode</code>: 指定运动模式, 0 为绝对运动, 1 为增量运动</p>
返回值	<p>成功: (0,)</p> <p>失败: 其它</p>

代码示例:

'''servo\_p 的使用'''

```

1.  # -*- coding: utf-8 -*-
2.  import sys
3.  sys.path.append('D:\\vs2019ws\\PythonCtt\\PythonCtt')
4.  import time
5.  import jkrc
6.  PI=3.1415926
7.
8.  ABS = 0          # 绝对运动
9.  INCR = 1         # 增量运动
10. Enable = True
11. Disable = False
12.
13. robot = jkrc.RC("192.168.2.160")#返回一个机器人对象
14. robot.login()#登录
15. robot.power_on() #上电
16. robot.enable_robot()
17. joint_pos=[PI/3,PI/3,-PI/3,PI/4,PI/4,0]
18. robot.joint_move(joint_pos,ABS,True,1)
19. robot.servo_move_enable(Enable) #进入位置控制模式
20. print("enable")
21. time.sleep(0.5)
22. for i in range(200):
23.     robot.servo_p(cartesian_pose = [1, 0, 0, 0, 0, 0],move_mode = INCR)
24.     time.sleep(0.008)
25. for i in range(200):
26.     robot.servo_p(cartesian_pose = [-1,0, 0, 0, 0, 0],move_mode = INCR)

```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```

27.     time.sleep(0.008)
28. robot.servo_move_enable(Disable)#退出位置控制模式
29. print("disable")
30. robot.logout() #登出

```

## 4.23 机器人笛卡尔空间伺服模式运动扩展

函数	<b>servo_p_extend(cartesian_pose, move_mode, step_num)</b>
描述	<p>(a) 机器人笛卡尔空间位置控制模式,需要注意的事项如下:</p> <p>(b) 当用户使用这个接口的时候需要先调用 servo_move_enable(True), 进入位置控制模式</p> <p>(c) 这条指令一般是高校科研当中做轨迹规划时使用。</p> <p>(d) 当用户使用此模式控制机器人的运动的时候, 控制器的规划器不参与运动的插补, 控制器进行逆解后, 会将位置指令会直接发给伺服, 所以用户需要自己进行轨迹规划, 否则机器人的运动效果会比较差, 如抖动剧烈等现象, 无法达到用户的预期。</p> <p>(e) 由于控制器的控制周期为 8ms, 建议用户的发送周期也为 8ms, 并且需要连续发送, 只发一次没有效果。如果网络状况较差, 发送周期可以小于 8ms。</p> <p>(f) Jaka 机器人的关节速度上限是 180 度/秒, 如果发送的位置使得机器人的关节速度超过了此上限。这条指令就会失效。</p> <p>(g) 使用完这条指令, 需要使用 servo_move_enable(False),退出位置控制模式。</p> <p>(h) 这条指令和前文提到的 linear_move()区别较大, linear_move 的插补是由控制器进行的, 用户无需关心。用户使用 servo_p 指令时需要预先进行轨迹的规划, 否则使用效果会很差, 无法达到预期。无特殊需要, 机器人在笛卡尔空间的运动建议使用 linear_move, 而不是 servo_p。</p>
参数	<p>cartesian_pose: 机器人笛卡尔空间运动目标位置</p> <p>move_mode: 指定运动模式, 0 为绝对运动, 1 为增量运动</p> <p>step_num: 倍分周期, servo_p 运动周期为 step_num*8ms, 其中 step_num&gt;=1</p>
返回值	<p>成功: (0,)</p> <p>失败: 其它</p>

## 4.24 设置数字输出变量(DO)的值

函数	<b>set_digital_output(iotype = a_type, index = a_number, value = a_value)</b>
描述	设置数字输出变量(DO)的值
参数	<p>iotype : DO 类型</p> <p>index : DO 索引</p> <p>value : DO 设置值</p>
返回值	<p>成功: (0,)</p> <p>失败: 其它</p>

代码示例:

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

'''设置 DO3 的值为 1'''

```
1. # -*- coding: utf-8 -*-
2. import sys
3. sys.path.append('D:\\vs2019ws\\PythonCtt\\PythonCtt')
4. import time
5. import jkrc
6. PI=3.1415926
7.
8. IO_CABINET = 0 #控制柜面板 IO
9. IO_TOOL = 1 #工具 IO
10. IO_EXTEND = 2 #扩展 IO
11.
12. robot = jkrc.RC("192.168.2.160")
13. robot.login()
14. ret = robot.get_digital_output(0,2)
15. if ret[0] == 0:
16.     print("1the DO2 is :",ret[1])
17. else:
18.     print("some things happend,the errcode is: ",ret[0])
19. robot.set_digital_output(IO_CABINET, 2, 1)#设置 DO2 的引脚输出值为 1
20. time.sleep(0.1)
21. ret = robot.get_digital_output(0, 2)
22. if ret[0] == 0:
23.     print("2the DO2 is :",ret[1])
24. else:
25.     print("some things happend,the errcode is: ",ret[0])
26. robot.logout() #登出
```

## 4.25 设置模拟输出变量的值(AO)的值

函数	set_analog_output(iotype = a_type, index = a_number, value = a_value)
描述	设置模拟输出变量的值(AO)的值
参数	iotype : AO 类型 index : AO 索引 value : AO 设置值
返回值	成功: (0,) 失败: 其它

代码示例:

'''设置 AO4 的值为 1.55'''

```
import jkrc
IO_CABINET = 0 #控制柜面板 IO
IO_TOOL = 1 #工具 IO
IO_EXTEND = 2 #扩展 IO
```

**上海节卡机器人科技有限公司** Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```
robot = jkrc.RC("192.168.2.64")
robot.login()
robot.set_analog_output(iotype = IO_CABINET, index = 3, value = 1.55) #
robot.logout() # 登出
```

## 4.26 查询数字输入(DI)状态

函数	get_digital_input(iotype, index)
描述	查询数字输入(DI)状态
参数	iotype: DI 类型 index: DI 索引
返回值	成功: (0, value), value: DI 状态查询结果 失败: 其它

## 4.27 查询数字输出(DO)状态

函数	get_digital_output(iotype, index)
描述	查询数字输出(DO)状态
参数	iotype: DO 类型 index: DO 索引
返回值	成功: (0, value), value: DO 状态查询结果 失败: 其它

## 4.28 查询模拟量输入变量(AI)的值

函数	get_analog_input(iotype, index)
描述	获取模拟量输入变量(AI)的值
参数	iotype : AI 类型 index : AI 索引
返回值	成功: (0, value), value 是 AI 状态查询结果, 为浮点数 失败: 其它

## 4.29 查询模拟量输出变量(AO)的值

函数	get_analog_output(type, index)
描述	获取模拟量输出变量(AO)的值
参数	type : AO 类型 index : AO 索引
返回值	成功: (0, value), value 是 AO 状态查询结果, 为浮点数



失败：其它
-------

代码示例：

```
''' 查询 AO4 的值为'''
import jkrc
IO_CABINET = 0 #控制柜面板 IO
IO_TOOL = 1 #工具 IO
IO_EXTEND = 2 #扩展 IO
robot = jkrc.RC("192.168.2.64")
robot.login()
robot.set_analog_output(iotype = IO_CABINET,index = 3,value = 1.55)
ret = robot.get_analog_output(iotype = IO_CABINET,index = 3)
print("AO value is:",ret[1])
robot.logout() #登出
```

## 4.30 查询扩展 IO 模块是否运行

函数	is_extio_running()
描述	查询扩展 IO 模块是否运行
参数	
返回值	成功：(0, status)，status 为 1 时代表扩展 IO 模块，为 0 时则相反 失败：其它

## 4.31 加载指定的作业程序

函数	program_load(file_name)
描述	加载指定的作业程序
参数	file_name：程序名称，传入的名称是一个字符串，比如“file_name”
返回值	成功：(0,) 失败：其它

## 4.32 获取已加载的作业程序名称

函数	get_loaded_program()
描述	获取已加载的作业程序名称
参数	
返回值	成功：(0, file_name) 失败：其它

代码示例：

```
import jkrc
```

**上海节卡机器人科技有限公司** Shanghai JAKA Robotics Ltd

电话 Tel：+400 006 2665 | 网站 Web:www.jaka.com

上海：上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州：江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu



```
robot = jkrc.RC("192.168.2.64") # 返回一个机器人对象
robot.login() # 登录
ret = robot.program_load("program_test") # 加载通过 APP 编写的脚本 program_test 需要自己编写
ret = robot.get_loaded_program()
print("the loaded program is:", ret[1])
robot.logout() # 登出
```

### 4.33 获取当前机器人作业程序的执行行号

函数	<b>get_current_line()</b>
描述	获取当前机器人作业程序的执行行号
参数	
返回值	成功: (0, curr_line), curr_line: 当前行号查询结果。 失败: 其它

### 4.34 运行当前加载的作业程序

函数	<b>program_run()</b>
描述	运行当前加载的作业程序
参数	
返回值	成功: (0,) 失败: 其它

### 4.35 暂停当前运行的作业程序

函数	<b>program_pause()</b>
描述	暂停当前运行的作业程序
参数	
返回值	成功: (0,) 失败: 其它

### 4.36 继续运行当前暂停的作业程序

函数	<b>program_resume()</b>
描述	继续运行当前暂停的作业程序
参数	
返回值	成功: (0,) 失败: 其它

## 4.37 终止当前执行的作业程序

函数	<b>program_abort()</b>
描述	终止当前执行的作业程序
参数	
返回值	成功: (0,) 失败: 其它

## 4.38 获取机器人作业程序执行状态

函数	<b>get_program_state()</b>
描述	获取机器人作业程序执行状态
参数	
返回值	成功: (0, state), state 的值有三种可能分别是 0、1、2 0 代表程序停止运行 1 代表程序正在运行 2 代表程序暂停 失败: 其它

代码示例:

```
1. # -*- coding: utf-8 -*-
2. import sys
3. sys.path.append('D:\\vs2019ws\\PythonCtt\\PythonCtt')
4. import time
5. import jkrc
6. import _thread
7. PI=3.1415926
8.
9. def print_state(name,robot):
10.     while(1):
11.         ret = robot.get_program_state() #查询程序运行状态,0 程序终止或无程序运行,1 程序运行中,
            2 暂停
12.         print("the robot program state is:",ret[1])
13.         time.sleep(1)
14.
15. robot = jkrc.RC("192.168.2.160")#返回一个机器人对象
16. robot.login() #登录
17. ret = robot.program_load("simple")#加载通过 APP 编写的脚本 program_test 需要自己编写
18. ret = robot.get_loaded_program()
19. print("the loaded program is:",ret[1])
20. robot.program_run()
21. _thread.start_new_thread( print_state,("p1_state", robot))#开启一个“p1”线程查询程序状态
```

```

22. time.sleep(10)
23. robot.program_pause() #暂停
24. time.sleep(10)
25. robot.program_resume() #恢复
26. time.sleep(10)
27. robot.program_abort() #停止
28. time.sleep(3)
29. robot.logout() #登出

```

## 4.39 设置机器人运行速度倍率

函数	<b>set_rapidrate(rapid_rate)</b>
描述	设置机器人运行速度倍率
参数	rapid_rate: 机器人运行速度倍率
返回值	成功: (0,) 失败: 其它

## 4.40 获取机器人运行速度倍率

函数	<b>get_rapidrate()</b>
描述	获取机器人运行速度倍率
参数	
返回值	成功: (0, rapidrate), rapidrate 是速度倍率, 返回值是 0 到 1 之间的闭区间 失败: 其它

## 4.41 设置指定编号的工具信息

函数	<b>set_tool_data(id, tcp, name)</b>
描述	设置指定编号的工具信息
参数	id: 设置工具 ID, 可选 ID 为 1 到 10, 0 代表末端法兰盘, 已被控制器使用。 tcp: 设置工具坐标系参数[x,y,z,rx,ry,rz] name: 用户坐标系别名
返回值	成功: (0,) 失败: 其它

## 4.42 设置当前使用的工具 ID

函数	<b>set_tool_id(id)</b>
描述	设置当前使用的工具 ID
参数	id: 工具坐标系 ID

返回值	成功: (0,) 失败: 其它
-----	--------------------

## 4.43 查询目标工具坐标系的信息

函数	<b>get_tool_data(id)</b>
描述	查询机器人使用的工具信息
参数	
返回值	成功: (0, (id, tcp)), id 值范围为 0 到 10, 0 代表末端法兰盘, 已被控制器使用。 tcp: 工具坐标系参数[x,y,z,rx,ry,rz] 失败: 其它

## 4.44 查询机器人当前使用的工具 ID

函数	<b>get_tool_id()</b>
描述	查询机器人当前使用的工具 ID
参数	
返回值	成功: (0, id), id 值范围为 0 到 10, 0 代表末端法兰盘, 已被控制器使用。 失败: 其它

代码示例:

```

1.  # -*- coding: utf-8 -*-
2.  import sys
3.  sys.path.append('D:\\vs2019ws\\PythonCtt\\PythonCtt')
4.  import time
5.  import jkrc
6.  PI=3.1415926
7.
8.  robot = jkrc.RC("192.168.2.160") #返回一个机器人对象
9.  robot.login()                  #登录
10. ret = robot.get_tool_data(1)   #查询工具坐标系数据
11. if ret[0] == 0:
12.     print("the tool data is :",ret)
13. else:
14.     print("some things happend,the errcode is: ",ret[0])
15. robot.set_tool_data(1,[0,0,1,0,0,0],'testlx') #设置工具坐标系数据
16. time.sleep(0.5)
17. ret = robot.get_tool_data(1)   #查询工具坐标系数据
18. if ret[0] == 0:
19.     print("the tool data is :",ret)

```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```

20. else:
21.     print("some things happend,the errcode is: ",ret[0])
22. ret = robot.get_tool_id()      #查询工具坐标系 id
23. print("tool_id",ret)
24. robot.set_tool_id(1)          #设置工具坐标系数据
25. time.sleep(0.5)
26. ret = robot.get_tool_id()      #查询工具坐标系 id
27. print("tool_id",ret)
28. robot.logout()

```

## 4.45 设置用户坐标系信息

函数	<b>set_user_frame_data(id, user_frame, name)</b>
描述	设置用户坐标系信息。
参数	id: 用户坐标系 ID, 可选 ID 为 1 到 10, 0 代表机器人基坐标系 user_frame: 用户坐标系参数[x,y,z,rx,ry,rz] name: 用户坐标系别名
返回值	成功: (0,) 失败: 其它

## 4.46 获取用户坐标系信息

函数	<b>get_user_frame_data()</b>
描述	获取用户坐标系信息
参数	
返回值	成功: (0, id, tcp), id: 用户坐标系 ID, 可选 ID 为 1 到 10, 0 代表机器人基坐标系 tcp: 用户坐标系参数[x,y,z,rx,ry,rz] 失败: 其它

## 4.47 设置当前使用的用户坐标系 ID

函数	<b>set_user_frame_id(id)</b>
描述	设置当前使用的用户坐标系 ID
参数	id: 用户坐标系 ID
返回值	成功: (0,) 失败: 其它

## 4.48 查询当前使用的用户坐标系 ID

函数	<code>get_user_frame_id()</code>
描述	查询当前使用的用户坐标系 ID
参数	
返回值	成功: (0, id), id 值范围为 0 到 10, 0 代表机器人基坐标系 失败: 其它

## 4.49 控制器机器人进入或退出拖拽模式

函数	<code>drag_mode_enable(enable)</code>
描述	控制器机器人进入或退出拖拽模式
参数	
返回值	成功: (0,) 失败: 其它

代码示例:

```
1. import jkrc
2. import time
3. #坐标系
4. COORD_BASE = 0
5. COORD_JOINT = 1
6. COORD_TOOL = 2
7. #运动模式
8. ABS = 0
9. INCR = 1
10. robot = jkrc.RC("192.168.2.160")
11. robot.login()
12. robot.power_on()
13. robot.enable_robot()
14. robot.drag_mode_enable(True)
15. ret = robot.is_in_drag_mode()
16. print(ret)
17. a = input()
18. robot.drag_mode_enable(False)
19. ret = robot.is_in_drag_mode()
20. print(ret)
21. robot.logout()
```

## 4.50 查询机器人是否处于拖拽模式

函数	<b>is_in_drag_mode()</b>
描述	查询机器人是否处于拖拽模式
参数	
返回值	成功: (0,state) state 为 1 代表机器人处于拖拽模式, 0 则相反 失败: 其它

## 4.51 获取机器人状态

函数	<b>get_robot_state()</b>
描述	获取机器人状态
参数	
返回值	成功: (0, state), state 是一个包含 3 位元素的元组, 分别代表是否急停, 是否打开电源, 是否上伺服使能。1 代表是, 0 代表否。 失败: 其它

代码示例:

```
1. # -*- coding: utf-8 -*-
2. import sys
3. #sys.path.append('D:\\vs2019ws\\PythonCtt\\lib')
4. import time
5. import jkrc
6.
7. robot = jkrc.RC("192.168.2.160")#返回一个机器人对象
8. ret = robot.login()#登录
9.
10. ret = robot.get_robot_state()
11. if ret[0] == 0:
12.     print("the robot state is :",ret[1])
13. else:
14.     print("some things happend,the errcode is: ",ret[0])
15. robot.logout() #登出
```

## 4.52 查询机器人是否超出限位

函数	<b>is_on_limit()</b>
描述	查询机器人是否超出限位
参数	
返回值	成功: (0, state), state 为 1 代表机器人超出限位, 0 则相反 失败: 其它

## 4.53 查询机器人运动是否停止

函数	<code>is_in_pos()</code>
描述	查询机器人运动是否停止
参数	
返回值	成功: (0, state), state 为 1 代表机器人停止, 0 则相反 失败: 其它

## 4.54 查询机器人是否处于碰撞保护模式

函数	<code>is_in_collision()</code>
描述	查询机器人是否处于碰撞保护模式
参数	
返回值	成功: (0, state), state 为 1 代表机器人处于碰撞保护模式, 0 则相反 失败: 其它

## 4.55 碰撞之后从碰撞保护模式恢复

函数	<code>collision_recover()</code>
描述	碰撞之后从碰撞保护模式恢复
参数	
返回值	成功: (0,) 失败: 其它

代码示例:

```
1. from typing import Counter
2. import jkrc
3. import time
4.
5.
6. robot = jkrc.RC("192.168.2.160")
7. robot.login()
8. robot.power_on()
9. robot.enable_robot()
10. ret = robot.get_collision_level()#获取当前碰撞等级
11. print(ret)
```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu



```

12. robot.set_collision_level(1)#设置碰撞等级
13. ret = robot.get_collision_level()
14. print(ret)
15. num = 0
16. while(1):
17.     ret = robot.is_in_collision() #查询是否处于碰撞保护模式
18.     collision_status = ret[1]
19.     if collision_status == 1:
20.         time.sleep(5)
21.         robot.collision_recover() #如果发生了碰撞，从碰撞保护模式恢复
22.         print(" in collision "+ str(num))
23.     else:
24.         print("the robot is not in collision "+ str(num))
25.         time.sleep(1)
26.         num=num+1
27.
28. robot.logout()

```

## 4.56 设置机器人碰撞等级

函数	<b>set_collision_level(level)</b>
描述	设置机器人碰撞等级
参数	level: 碰撞等级，等级 0-5，0 为关闭碰撞，1 为碰撞阈值 25N，2 为碰撞阈值 50N，3 为碰撞阈值 75N，4 为碰撞阈值 100N，5 为碰撞阈值 125N
返回值	成功：(0,) 失败：其它

## 4.57 获取机器人碰撞等级

函数	<b>get_collision_level()</b>
描述	获取机器人碰撞等级
参数	
返回值	成功：(0, level)，level 是返回的碰撞等级，等级 0-5。 0 为关闭碰撞， 1 为碰撞阈值 25N， 2 为碰撞阈值 50N， 3 为碰撞阈值 75N， 4 为碰撞阈值 100N， 5 为碰撞阈值 125N 失败：其它

## 4.58 欧拉角到旋转矩阵的转换

函数	<b>ropy_to_rot_matrix(ropy = [rx,ry,rz])</b>
描述	欧拉角到旋转矩阵的转换
参数	ropy:待转换的欧拉角数据[rx,ry,rz]
返回值	成功: (0, rot_matrix), rot_matrix 是一个 3x3 的旋转矩阵 失败: 其它

## 4.59 旋转矩阵到欧拉角的转换

函数	<b>rot_matrix_to_ropy(rot_matrix)</b>
描述	旋转矩阵到欧拉角的转换
参数	rot_matrix: 转换的旋转矩阵数据
返回值	成功: (0, rpy), rpy 是长度为 3 的欧拉角元组(rx,ry,rz) 失败: 其它

## 4.60 四元数到旋转矩阵的转换

函数	<b>quaternion_to_rot_matrix (quaternion = [w,x,y,z])</b>
描述	四元数到旋转矩阵的转换
参数	quaternion: 待转换的四元数数据
返回值	成功: (0, rot_matrix), rot_matrix 是一个 3x3 的旋转矩阵 失败: 其它

## 4.61 旋转矩阵到四元数的转换

函数	<b>rot_matrix_to_quaternion(rot_matrix)</b>
描述	旋转矩阵到四元数的转换
参数	rot_matrix: 3x3 的旋转矩阵
返回值	成功: (0, quaternion), quaternion 是长度为 4 的四元数元组(w,x,y,z) 失败: 其它

代码示例:

```

1. import jkrc
2. robot = jkrc.RC("192.168.2.160")
3. robot.login()
4. ret = robot.get_tcp_position()
5. print(ret)
6. rpy = [ret[1][3], ret[1][4], ret[1][5]]#获取 rpy
7. ret = robot.ropy_to_rot_matrix(rpy)#rpy 转换成旋转矩阵

```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```

8. print(ret)
9. rot_matrix = ret[1] #获取旋转矩阵
10. ret = robot.rot_matrix_to_rpy(rot_matrix) #旋转矩阵转换成 rpy
11. print(ret)
12. ret = robot.rot_matrix_to_quaternion(rot_matrix) #旋转矩阵转换成四元数
13. print(ret)
14. quaternion = ret[1]
15. ret = robot.quaternion_to_rot_matrix(quaternion) #旋转矩阵转换成四元数
16. print(ret)
17. robot.logout()

```

## 4.62 注册机器人出现错误时的回调函数

函数	<b>set_error_handler(func)</b>
描述	注册机器人出现错误时的回调函数
参数	func:用户定义的回调函数
返回值	成功: (0,) 失败: 其它

## 4.63 机器人负载设置

函数	<b>set_payload(mass = m, centroid = [x,y,z])</b>
描述	机器人负载设置
参数	mass: 负载质量, 单位: kg centroid: 负载质心坐标[x, y ,z], 单位: mm
返回值	成功: (0,) 失败: 其它

代码示例:

```

import jkrc #导入模块
robot = jkrc.RC("192.168.2.226") #返回一个机器人对象
robot.login()
robot.set_payload(mass= 1, centroid =[0.01,0.02,0.03])
robot.logout() #登出

```

## 4.64 获取机器人负载数据

函数	<b>get_payload()</b>
描述	获取机器人负载数据

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

参数	
返回值	成功: (0, payload), payload 是长度为 2 的元组 (mass, (x,y,z)), 第一个元素是负载质量, 第二个元素是质心坐标。 失败: 其它

代码示例:

```

1.  ''' 设置负载'''
2.  import jkrc
3.  robot = jkrc.RC("192.168.2.64") #返回一个机器人对象
4.  robot.login() #登录
5.  robot.set_payload(mass= 1,centroid =[0.01,0.02,0.03])
6.  ret = robot.get_payload()
7.  if ret[0] == 0:
8.      print("the payload is :",ret[1])
9.  else:
10.     print("some things happend,the errcode is: ",ret[0])
11. robot.logout()

```

## 4.65 终止当前机械臂运动

函数	<b>motion_abort ()</b>
描述	可以在任何情况下终止机器人的运动
参数	
返回值	成功: (0,) 失败: 其它

代码示例:

```

1.  # -*- coding: utf-8 -*-
2.  import sys
3.  #sys.path.append('D:\\vs2019ws\\PythonCtt\\lib')
4.  import time
5.  import jkrc
6.  robot = jkrc.RC("192.168.2.160")#返回一个机器人对象
7.  robot.login()#登录
8.  robot.power_on() #上电
9.  robot.enable_robot()
10. print("move1")
11. robot.joint_move(joint_pos=[1,0,0,0,0,0],move_mode=1,is_block=False,speed=0.05)#增量运动
12. print("wait")
13. time.sleep(2)
14. print("stop")
15. robot.motion_abort()
16. robot.logout()#登出

```

## 4.66 获取机器人状态监测数据

函数	<code>get_robot_status ()</code>
描述	获取机器人状态数据监测数据
参数	
返回值	<p>成功: <code>(0, robotstatus)</code>, <code>robotstatus</code> 的长度为 24, <code>robotstatus</code> 返回数据顺序如下所示:</p> <ol style="list-style-type: none"> <li>1. <code>errcode</code> 机器人运行出错时错误编号, 0 为运行正常, 其它为异常</li> <li>2. <code>inpos</code> 机器人运动是否到位标志, 0 为没有到位, 1 为运动到位</li> <li>3. <code>powered_on</code> 机器人是否上电标志, 0 为没有上电, 1 为上电</li> <li>4. <code>enabled</code> 机器人是否使能标志, 0 为没有使能, 1 为使能</li> <li>5. <code>rapidrate</code> 机器人运行倍率</li> <li>6. <code>protective_stop</code> 机器人是否检测到碰撞, 0 为没有检测到碰撞, 1 则相反</li> <li>7. <code>drag_status</code> 机器人是否处于拖拽状态, 0 为没有处于拖拽状态, 1 则相反</li> <li>8. <code>on_soft_limit</code> 机器人是否处于限位, 0 为没有触发限位保护, 1 为触发限位保护</li> <li>9. <code>current_user_id</code> 机器人目前使用的用户坐标系 id</li> <li>10. <code>current_tool_id</code> 机器人目前使用的工具坐标系 id</li> <li>11. <code>dout</code> 机器人控制柜数字输出信号</li> <li>12. <code>din</code> 机器人控制柜数字输入信号</li> <li>13. <code>aout</code> 机器人控制柜模拟输出信号</li> <li>14. <code>ain</code> 机器人控制柜模拟输入信号</li> <li>15. <code>tio_dout</code> 机器人末端工具数字输出信号</li> <li>16. <code>tio_din</code> 机器人末端工具数字输入信号</li> <li>17. <code>tio_ain</code> 机器人末端工具模拟输入信号</li> <li>18. <code>extio</code> 机器人外部扩展模块 IO 信号</li> <li>19. <code>cart_position</code> 机器人末端的笛卡尔空间位置值</li> <li>20. <code>joint_position</code> 机器人关节空间位置</li> <li>21. <code>robot_monitor_data</code> 机器人状态监测数据 (scb 主版本号、scb 小版本号、控制器温度、机器人平均电压、机器人平均电流、机器人 6 个关节的监测数据 (瞬时电流、瞬时电压、瞬时温度))</li> <li>22. <code>torq_sensor_monitor_data</code> 机器人力矩传感器状态监测数据 (力矩传感器 ip 地址、力矩传感器端口号、工具负载 (负载质量、质心 x 轴坐标、质心 y 轴坐标、质心 z 轴坐标)、力矩传感器状态、力矩传感器异常错误码、6 个力矩传感器实际接触力值、6 个力矩传感器原始读数)</li> <li>23. <code>is_socket_connect sdk</code> 与控制器连接通道是否正常, 0 为连接通道异常, 1 为连接通道正常</li> <li>24. <code>emergency_stop</code> 机器人是否急停, 0 为没有按下急停, 1 则相反</li> </ol> <p>失败: 其它</p> <p>说明:</p> <p>若机器人无对应 IO, 会返回如“no extio”字符串。</p> <p>错误码文件可以在 JAKA 二次开发资料包\参考资料中找到</p>

代码示例:

```
1. import sys
```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```

2. import jkrc
3. import time
4.
5. robot = jkrc.RC("192.168.2.160")
6. robot.login()
7. ret = robot.get_robot_status()
8. if ret[0] == 0:
9.     print("the joint position is :",ret[1])
10.    print(len(ret[1]))
11.    for i in range(len(ret[1])):
12.        print(ret[1][i])
13. else:
14.     print("some things happend,the errcode is: ",ret[0])
15. robot.logout()

```

## 4.67 设置 SDK 是否开启调试模式

函数	set_debug_mode(mode)
描述	设置 SDK 是否开启调试模式。
参数	mode: 选择 TRUE 时，开始调试模式，此时会在标准输出流中输出调试信息，选择 FALSE
返回值	成功: (0,) 失败: 其它

## 4.68 设置错误码文件的路径

函数	set_errcode_file_path (errcode_file_path)
描述	设置错误码文件路径，使用 get_last_error 接口时，如果要获得具体的错误信息，需要使用此接口设置错误码文件路径，如果只是需要获取错误码，不需要调用此接口。 <b>注:路径不能包含中文否则无法使用</b>
参数	errcode_file_path: 错误码文件存放路径，错误码文件可以在 JAKA 二次开发资料包\参考资料中找到
返回值	成功: (0,) 失败: 其它

## 4.69 获取最新的错误码

函数	get_last_error ()
描述	获取机器人运行过程中最后一个错误码,当调用 clear_error 时，最后一个错误码会清

	零。如果要获得具体的错误信息，需要使用 <code>set_errorcode_file_path</code> 设置错误码文件路径，如果只是需要获取错误码，不需要调用 <code>set_errorcode_file_path</code> 。
参数	
返回值	成功: (0, error) 失败: 其它

代码示例:

```

1. import jkrc
2. robot = jkrc.RC("192.168.2.194")#返回一个机器人对象
3. robot.login() #登录
4. robot.program_load("not_exist") #故意加载一个不存在的程序,引起报错
5. ret = robot.get_last_error ()#没有设置错误码文件路径，只能得到错误码，得不到具体错误信息
6. print(ret[1])
7. robot.set_errorcode_file_path("D:\\JAKA_ERROR_CODE.csv") #路径不能包含中文
8. ret = robot.get_last_error () #设置错误码文件路径后，能得到错误码和具体错误信息
9. print(ret[1])
10. robot.logout() #登出

```

## 4.70 错误状态清除

函数	<code>clear_error()</code>
描述	错误状态清除。
参数	
返回值	成功: (0,) 失败: 其它

代码示例:

参考碰撞保护模式恢复

## 4.71 设置轨迹复现配置参数

函数	<code>set_traj_config(xyz_interval, rpy_interval, vel, acc)</code>
描述	设置轨迹复现配置参数，可设置空间位置采集精度，姿态采集精度，执行脚本运行速度，执行脚本运行加速度。
参数	xyz_interval: 空间位置采集速度 rpy_interval: 姿态采集精度 vel: 执行脚本运行速度 acc: 执行脚本运行加速度
返回值	成功: (0,) 失败: 其它

代码示例:

```

1. import jkrc
2. import time
3.

```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```

4. #坐标系
5. COORD_BASE = 0
6. COORD_JOINT = 1
7. COORD_TOOL = 2
8. #运动模式
9. ABS = 0
10. INCR= 1
11.
12. robot = jkrc.RC("192.168.2.160")
13. robot.login()
14. robot.power_on()
15. robot.enable_robot()
16. robot.joint_move(joint_pos =[1,1,1,1,1,1] ,move_mode = 0 ,is_block = True ,speed = 10 )
17. print("joint")
18. robot.set_traj_config([0.1, 0.1, 25, 100]) #设置轨迹复现参数, 仅录制过程有效
19. time.sleep(0.1)
20. ret = robot.get_traj_config()#获取轨迹复现参数
21. print("traj_config:")
22. print(ret)
23. robot.set_traj_sample_mode(True, 'pythonTrack')#开启轨迹复现采集
24. time.sleep(0.1)
25. robot.joint_move(joint_pos =[-1,1,1,1,1,1] ,move_mode = 0 ,is_block = True ,speed = 30*3.1
    4/180 )#阻塞运动
26. robot.joint_move(joint_pos =[1,1,1,1,1,1] ,move_mode = 0 ,is_block = True ,speed = 30*3.14
    /180 )
27. # robot.jog(2,INCR,COORD_BASE,10,-2)
28. # robot.jog(2,INCR,COORD_BASE,10,2)
29. robot.set_traj_sample_mode(False, 'pythonTrack')#结束轨迹复现采集
30. time.sleep(1)
31. res = robot.generate_traj_exe_file('pythonTrack')#将采集到的轨迹复现文件转化为可执行脚本
32. print(res)
33. robot.program_load("track/pythonTrack")#加载轨迹程序
34. time.sleep(0.1)
35. robot.program_run()

```

## 4.72 获取轨迹复现配置参数

函数	get_traj_config()
描述	获取轨迹复现配置参数, 可获得空间位置采集精度, 姿态采集精度, 执行脚本运行速度, 执行脚本运行加速度。
参数	
返回值	成功: (0, (xyz_interval, rpy_interval, vel, acc)) xyz_interval: 空间位置采集速度



	rpy_interval: 姿态采集精度 vel: 执行脚本运行速度 acc: 执行脚本运行加速度 失败: 其它
--	---

代码示例:

## 4.73 采集轨迹复现数据控制开关

函数	<b>set_traj_sample_mode(mode, filename)</b>
描述	采集轨迹复现数据控制开关。
参数	mode: 控制模式, True 代表开始采集数据, False 代表结束数据采集。 filename: 数据的存储文件名。
返回值	成功: (0,) 失败: 其它

代码示例:

## 4.74 采集轨迹复现数据状态查询

函数	<b>get_traj_sample_status()</b>
描述	采集轨迹复现数据状态查询。注: 在数据采集状态时, 不允许再次开启数据采集开关。
参数	
返回值	成功: (0, sample_status), sample_status: 数据状态, True 代表正在采集数据, False 代表数据采集结束。 失败: 其它

代码示例:

## 4.75 查询控制器中已经存在的轨迹复现数据的文件名

函数	<b>get_exist_traj_file_name()</b>
描述	查询控制器中已经存在的轨迹复现数据的文件名
参数	
返回值	成功: (0,) 失败: 其它

代码示例:

## 4.76 重命名轨迹复现数据的文件名

函数	<b>rename_traj_file_name(src, dest)</b>
描述	重命名轨迹复现数据的文件名
参数	src: 源文件名 dest: 目标文件名, 文件名长度不能超过 100 个字符, 不能为空, 目标文件名不支持中文。
返回值	成功: (0,) 失败: 其它

代码示例:

```

1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.226")          #返回一个机器人对象
3. robot.login()
4. robot.rename_traj_file_name('\\home\\src', '\\home\\dst')
5. robot.logout() #登出

```

## 4.77 删除控制器中轨迹复现数据的文件

函数	<b>remove_traj_file(filename)</b>
描述	删除控制器中轨迹复现数据文件
参数	filename: 所要删除的文件名
返回值	成功: (0,) 失败: 其它

代码示例:

```

1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.226")          #返回一个机器人对象
3. robot.login()
4. robot.remove_traj_file('test')
5. robot.logout() #登出

```

## 4.78 控制器中轨迹复现数据文件生成控制器执行脚本

函数	<b>generate_traj_exe_file(filename)</b>
描述	控制器中轨迹复现数据文件生成控制器执行脚本
参数	filename: 数据的文件名
返回值	成功: (0,) 失败: 其它

## 4.79 机器人扩展关节运动

函数	<b>joint_move_extend(joint_pos, move_mode, is_block, speed, acc, tol)</b>
描述	机器人扩展关节运动。增加关节角加速度和关节运动终点误差。
参数	<p>joint_pos: 机器人关节运动各目标关节角度。</p> <p>move_mode: 指定运动模式, 0 为绝对运动, 1 为增量运动, 2 代表连续运动。</p> <p>is_block: 是否为阻塞接口, True 为阻塞接口, False 为非阻塞接口。</p> <p>speed: 机器人关节运动速度, 单位: rad/s</p> <p>acc: 机器人关节运动角加速度。</p> <p>tol: 机器人运动终点误差。</p>
返回值	<p>成功: (0,)</p> <p>失败: 其它</p>

代码示例:

```

1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.226")          #返回一个机器人对象
3. robot.login()
4. robot.joint_move_extend(joint_pos=[1, 1, 1, 1, 1, 1],move_mode=0, is_block=True, speed=20,
    acc=5, tol=0.1)
5. robot.joint_move_extend(joint_pos=[-1, 1, 1, 1, 1, 0],move_mode=0, is_block=True, speed=20,
    acc=5, tol=0.1)
6. robot.logout() #登出

```

## 4.80 机器人扩展末端直线运动

函数	<b>linear_move_extend(end_pos, move_mode, is_block, speed, acc, tol)</b>
描述	机器人扩展末端直线运动。增加空间加速度和空间运动终点误差。
参数	<p>end_pos: 机器人末端运动目标位置。</p> <p>move_mode: 指定运动模式, 0 为绝对运动, 1 为增量运动</p> <p>is_block: 是否为阻塞接口, True 为阻塞接口, False 为非阻塞接口。</p> <p>speed: 机器人笛卡尔空间运动速度, 单位: mm/s</p> <p>acc: 机器人笛卡尔空间加速度, 单位: mm/s^2。</p> <p>tol: 机器人运动终点误差。</p>
返回值	<p>成功: (0,)</p> <p>失败: 其它</p>

## 4.81 机器人末端圆弧运动

函数	<b>circular_move(end_pos, mid_pos, move_mode, is_block, speed, acc, tol)</b>
描述	机器人末端圆弧运动
参数	end_pos: 机器人末端运动目标位置。

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

	mid_pos: 机器人末端运动中间点。 move_mode: 指定运动模式, 0 为绝对运动, 1 为增量运动, 2 代表连续运动。 is_block: 是否为阻塞接口, True 为阻塞接口, False 为非阻塞接口。 speed: 机器人关节运动速度, 单位: rad/s acc: 机器人关节运动角加速度。 tol: 机器人运动终点误差。
返回值	成功: (0,) 失败: 其它

代码示例:

```

1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.226") #返回一个机器人对象
3. robot.login()
4. robot.circular_move(end_pos = [100, 100, 0, 0, 1, 1], mid_pos = [100, 0, 0, 0, 1, 0], move_m
   ode= 0, is_block = True, speed = 20, acc = 5, tol = 0.1)
5. robot.logout() #登出

```

## 4.82 机器人 SERVO 模式下禁用滤波器

函数	servo_move_use_none_filter()
描述	机器人 SERVO 模式下禁用滤波器, 该指令在 SERVO 模式下不可设置, 退出 SERVO 后可设置。
参数	
返回值	成功: (0,) 失败: 其它

代码示例:

```

1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.226") #返回一个机器人对象
3. robot.login()
4. robot.servo_move_use_none_filter()
5. robot.logout() #登出

```

## 4.83 机器人 SERVO 模式下关节空间一阶低通滤波

函数	servo_move_use_joint_LPF(cutoffFreq)
描述	机器人 SERVO 模式下关节空间一阶低通滤波, 该指令在 SERVO 模式下不可设置, 退出 SERVO 后可设置。
参数	cutoffFreq: 一阶低通滤波器截止频率。
返回值	成功: (0,) 失败: 其它

代码示例:

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web: www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```

1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.226") #返回一个机器人对象
3. robot.login()
4. robot.servo_move_use_joint_LPF(0.5)
5. robot.logout() #登出

```

## 4.84 机器人 SERVO 模式下关节空间非线性滤波

函数	<b>servo_move_use_joint_NLF(max_vr, max_ar, max_jr)</b>
描述	机器人 SERVO 模式下关节空间非线性滤波，该指令在 SERVO 模式下不可设置，退出 SERVO 后可设置。
参数	max_vr: 笛卡尔空间姿态变化速度的速度上限值（绝对值）°/s max_ar: 笛卡尔空间姿态变化速度的加速度上限值（绝对值）°/s <sup>2</sup> max_jr: 笛卡尔空间姿态变化速度的加加速度上限值（绝对值）°/s <sup>3</sup>
返回值	成功: (0,) 失败: 其它

代码示例：

```

1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.226") #返回一个机器人对象
3. robot.login()
4. robot.servo_move_use_joint_NLF(max_vr=2, max_ar=2, max_jr=4)
5. robot.logout() #登出

```

## 4.85 机器人 SERVO 模式下笛卡尔空间非线性滤波

函数	<b>servo_move_use_carte_NLF(max_vp, max_ap, max_jp, max_vr, max_ar, max_jr)</b>
描述	机器人 SERVO 模式下笛卡尔空间非线性滤波，该指令在 SERVO 模式下不可设置，退出 SERVO 后可设置。
参数	max_vp: 笛卡尔空间下移动指令速度的上限值（绝对值）mm/s max_ap: 笛卡尔空间下移动指令加速度的上限值（绝对值）mm/s <sup>2</sup> max_jp: 笛卡尔空间下移动指令加加速度的上限值（绝对值）mm/s <sup>3</sup> max_vr: 笛卡尔空间姿态变化速度的速度上限值（绝对值）°/s max_ar: 笛卡尔空间姿态变化速度的加速度上限值（绝对值）°/s <sup>2</sup> max_jr: 笛卡尔空间姿态变化速度的加加速度上限值（绝对值）°/s <sup>3</sup>
返回值	成功: (0,) 失败: 其它

代码示例：

```

1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.226") #返回一个机器人对象
3. robot.login()

```

```

4. robot.servo_move_use_carte_NLF(max_vp=2, max_ap=2, max_jp=4,max_vr=2, max_ar=2, max_jr=4)
5. robot.logout() #登出

```

## 4.86 机器人 SERVO 模式下关节空间多阶均值滤波

函数	<b>servo_move_use_joint_MMF(max_buf, kp, kv, ka)</b>
描述	机器人 SERVO 模式下关节空间多阶均值滤波，该指令在 SERVO 模式下不可设置，退出 SERVO 后可设置。
参数	max_buf: 均值滤波器缓冲区的大小。 kp: 加速度滤波系数。 kv: 速度滤波系数 ka: 位置滤波系数
返回值	成功: (0,) 失败: 其它

代码示例：

```

1. import jkrc #导入模块
2. robot = jkrc.RC("192.168.2.226") #返回一个机器人对象
3. robot.login()
4. robot.servo_move_use_joint_MMF(max_buf=20 , kp=0.2 , kv=0.4 ,ka=0.2)
5. robot.logout() #登出

```

## 4.87 SERVO 模式下速度前瞻参数设置

函数	<b>servo_speed_foresight (max_buf, kp)</b>
描述	SERVO 模式下速度前瞻参数设置
参数	max_buf: 均值滤波器缓冲区的大小。 kp: 加速度滤波系数。
返回值	成功: (0,) 失败: 其它

## 4.88 设置 SDK 日志路径

函数	<b>set_SDK_filepath(filepath)</b>
描述	设置 SDK 日志路径
参数	filepath: 文件路径
返回值	成功: (0,) 失败: 其它

## 4.89 设置传感器品牌

函数	<code>set_torsensor_brand(sensor_brand)</code>
描述	设置传感器品牌，输入数字代表对应的传感器品牌，可选值为 1，2，3 分别代表不同的力矩传感器。
参数	<code>sensor_brand</code> : 传感器品牌所对应的数字
返回值	成功: (0,) 失败: 其它

代码示例:

```
1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.165") #返回一个机器人对象
3. robot.login()
4. robot.set_torsenosr_brand(1)    #1 为 SONY 索尼半导体;2 为 BoschSensortec 博世;3 为 ST 意法半导体
5. robot.logout() #登出
```

## 4.90 获取传感器品牌

函数	<code>get_torsensor_brand()</code>
描述	获取当前所使用的传感器品牌，输出的数字代表不同的传感器品牌。
参数	
返回值	成功: (0, sensor_brand), <code>sensor_brand</code> : 传感器品牌，可选值为 1，2，3 分别代表不同的力矩传感器。 失败: 其它

代码示例:

```
1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.165") #返回一个机器人对象
3. robot.login()
4. ret=robot.get_torsenosr_brand()
5. if ret[0] == 1:
6.     print("the sensor_band is SONY 索尼半导体")
7. elif ret[0] == 2:
8.     print("the sensor_band is BoschSensortec 博世")
9. else:
10.    print("the sensor_band is ST 意法半导体")
11. robot.logout() #登出
```

## 4.91 开启或关闭力矩传感器

函数	<b>set_torque_sensor_mode(sensor_mode)</b>
描述	开启或关闭当前所使用的力矩传感器。
参数	sensor_mode: 传感器工作模式, 0 代表关闭力矩传感器, 1 代表开启力矩传感器。
返回值	成功: (0,) 失败: 其它

## 4.92 设置柔顺控制参数

函数	<b>set_admit_ctrl_config(axis, opt, ftUser, ftReboundFK, ftConstant, ftNormalTrack)</b>
描述	设置机器人柔顺控制时, 关节坐标轴编号, 柔顺方向, 阻尼力, 回弹力, 恒力, 法向跟踪等参数。
参数	axis: 代表笛卡尔空间坐标轴编号, 取值为: 0~5, 分别对应 fx fy fz mx my mz 方向 opt: 柔顺方向, 0 代表关闭, 1 代表启动。 ftUser: 阻尼力, 表示用户需使用多大的力让机器人沿着某一方向以最大速度运动。 ftReboundFK: 回弹力, 表示机器人回到初始状态的能力。 ftConstant: 恒力, 手动操作时全部设置为 0。 ftNormalTrack: 法向跟踪, 手动操作时全部设置为 0。
返回值	成功: (0,) 失败: 其它

## 4.93 开始辨识工具末端负载

函数	<b>start_torq_sensor_payload_identify(joint_pos)</b>
描述	开始辨识工具末端负载, 输入为一个元组含 6 个元素, 分别对应结束位置 6 个关节角。
参数	joint_pos: 使用力矩传感器进行自动负载辨识时的结束位置, 对应 6 个关节角
返回值	成功: (0,) 失败: 其它

代码示例:

```

1. # -*- coding: utf-8 -*-
2. import sys
3. #sys.path.append('D:\\vs2019ws\\PythonCtt\\lib')
4. import time
5. import jkrc
6. PI = 3.1415926
7.
8. robot = jkrc.RC("10.5.5.100")#返回一个机器人对象

```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu



```
9. ret = robot.login()#登录
10. ret = robot.power_on()
11. ret = robot.enable_robot()
12. robot.set_torsenosr_brand(2)
13. robot.set_torque_sensor_mode(1)
14. robot.set_compliant_type(1, 1)
15. print("inint sensor comple")
16. print("ready to run")
17. ret = robot.get_joint_position()
18. joint_pos_origin = ret[1]
19. joint_pos = ret[1]
20. print(joint_pos)
21. joint_pos[3] += PI / 4
22. if (joint_pos[3] > 265 * PI / 180):
23.     joint_pos[3] -= PI/2
24. joint_pos[4] += PI / 4
25. if (joint_pos[4] > 320 * PI / 180):
26.     joint_pos[4] -= PI/2
27. joint_pos[5] += PI / 4
28. if (joint_pos[5] > 265 * PI / 180):
29.     joint_pos[5] -= PI
30. print(joint_pos)
31. ret = robot.start_torq_sensor_payload_identify(joint_pos)
32. time.sleep(1)
33. flag = 1
34. while (1 == flag):
35.     ret = robot.get_torq_sensor_identify_staus()
36.     print(ret)
37.     time.sleep(1)
38.     flag = ret[1]
39. print("identy_finish")
40. ret = robot.get_torq_sensor_payload_identify_result()
41. print(ret)
42. ret = robot.set_torq_sensor_payload()
43. print(ret)
44. ret = robot.get_torq_sensor_payload_identify_result()
45. print(ret)
46. robot.joint_move(joint_pos_origin,0,1,10)
47. print("back")
48. robot.logout() #登出
```

## 4.94 获取末端负载辨识状态

函数	<code>get_torq_sensor_identify_status()</code>
描述	获取末端负载辨识状态
参数	<code>identify_status</code> : 辨识状态, 0 代表辨识完成, 1 代表未完成, 2 代表辨识失败。
返回值	成功: <code>(0,identify_status)</code> 失败: 其它

## 4.95 获取末端负载辨识结果

函数	<code>get_torq_sensor_payload_identify_result()</code>
描述	获取末端负载辨识结果, 输入为负载质量, 负载质心位置坐标
参数	<code>mass</code> : 负载质量, 单位: kg <code>centroid</code> : 负载质心位置[x, y, z], 单位: mm
返回值	成功: <code>(0,)</code> 失败: 其它

## 4.96 设置传感器末端负载

函数	<code>set_torq_sensor_tool_payload (mass, centroid)</code>
描述	设置传感器末端负载, 输入为负载质量, 负载质心位置坐标。
参数	<code>mass</code> : 负载质量, 单位: kg <code>centroid</code> : 负载质心坐标位置[x, y, z], 单位: mm
返回值	成功: <code>(0,)</code> 失败: 其它

代码示例:

```
1. import jkrc                                #导入模块
2. robot = jkrc.RC("192.168.2.226")           #返回一个机器人对象
3. robot.login()
4. robot.set_torq_sensor_tool_payload(mass= 1, centroid =[10,10,0])
5. robot.logout() #登出
```

## 4.97 获取传感器末端负载

函数	<code>get_torq_sensor_tool_payload ()</code>
----	--

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

描述	获取传感器末端负载质量，负载质心位置坐标。
参数	
返回值	成功：(0,(mass, centroid)), mass: 负载质量, 单位: kg, centroid: 负载质心位置[x, y, z], 单位: mm 失败：其它

## 4.98 力控导纳使能控制

函数	<code>enable_admittance_ctrl(enable_flag)</code>
描述	力控导纳使能控制
参数	<code>enable_flag</code> : 标志，0 为关闭力控拖拽使能，1 为开启力控导纳使能。
返回值	成功：(0,) 失败：其它

代码示例：

```

1. # -*- coding: utf-8 -*-
2. import sys
3. #sys.path.append('D:\\vs2019ws\\PythonCtt\\lib')
4. import time
5. import jkrc
6. PI=3.1415926
7.
8. robot = jkrc.RC("10.5.5.100")#返回一个机器人对象
9. ret = robot.login()#登录
10. ret = robot.power_on()
11. ret = robot.enable_robot()
12. robot.set_torsenosr_brand(2)
13. robot.set_torque_sensor_mode(1)
14. robot.set_compliant_type(1, 1)
15. print("inint sensor comple")
16. print("ready to run")
17. #设置柔顺控制参数
18. ret = robot.set_admit_ctrl_config(0, 0, 20, 5, 0, 0)
19. ret = robot.set_admit_ctrl_config(1, 0, 20, 5, 0, 0)
20. ret = robot.set_admit_ctrl_config(2, 1, 20, 10, 0, 0)
21. ret = robot.set_admit_ctrl_config(3, 0, 20, 5, 0, 0)
22. ret = robot.set_admit_ctrl_config(4, 0, 20, 5, 0, 0)
23. ret = robot.set_admit_ctrl_config(5, 0, 20, 5, 0, 0)
24. #设置力控拖拽使能，1 打开，0 关闭
25. ret = robot.enable_admittance_ctrl(1)
26. print("enable_admittance_ctrl open! ")
27. print("input any word to quit:")

```

```

28. a = input()
29. ret = robot.enable_admittance_ctrl(0)
30. ret = robot.set_admit_ctrl_config(2, 0, 20, 5, 0, 0)
31. robot.set_torque_sensor_mode(0)
32. robot.logout() #登出

```

## 4.99 设置力控类型和传感器初始化状态

函数	set_compliant_type(sensor_compensation, compliance_type)
描述	设置力控类型和传感器初始化状态
参数	sensor_compensation: 开启传感器补偿标志, 1 代表开启初始化, 0 代表不初始化 compliance_type: 力控类型, 0 代表不使用任何一种柔顺控制方法 1 代表恒力柔顺控制, 2 代表速度柔顺控制
返回值	成功: (0,) 失败: 其它

代码示例:

```

import jkrc #导入模块
robot = jkrc.RC("192.168.2.226") #返回一个机器人对象
robot.login()
robot.set_compliant_type(1,1)
robot.logout() #登出

```

## 4.100 获取力控类型和传感器初始化状态

函数	get_compliant_type()
描述	获取力控类型和传感器初始化状态
参数	
返回值	成功: (0, sensor_compensation, compliance_type) sensor_compensation: 开启传感器补偿标志, 1 代表开启初始化, 0 代表不初始化 compliance_type: 力控类型, 0 代表不使用任何一种柔顺控制方法 1 代表恒力柔顺控制, 2 代表速度柔顺控制 失败: 其它

## 4.101 获取力控柔顺控制参数

函数	get_admit_ctrl_config()
描述	获取力控柔顺控制参数, 获取 6 个关节所对应的柔顺方向, 阻尼力, 回弹力, 恒力, 法向跟踪。
参数	

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海: 上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州: 江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

返回值	成功: (0, [[opt, ftUser, ftReboundFK, ftConstant, ftNormalTrack], .....]) opt: 柔顺方向, 可选值 1 2 3 4 5 6 分别对应 fx fy fz mx my mz 方向, 0 代表没有勾选。 ftUser: 阻尼力, 表示用户需使用多大的力让机器人沿着某一方向以最大速度运动。 ftReboundFK: 回弹力, 表示机器人回到初始状态的能力。 ftConstant: 恒力, 手动操作时全部设置为 0。 ftNormalTrack: 法向跟踪, 手动操作时全部设置为 0。 失败: 其它
-----	--

## 4.102 设置力矩传感器 IP 地址

函数	<b>set_torque_sensor_comm(type, ip_addr, port)</b>
描述	设置力矩传感器 IP 地址。
参数	type: 传感器类型 ip_addr: 传感器 ip 地址 port: 端口号
返回值	成功: (0,) 失败: 其它

## 4.103 获取力矩传感器 IP 地址

函数	<b>get_torque_sensor_comm()</b>
描述	获取力矩传感器 IP 地址。
参数	
返回值	成功: (0, ip_addr), ip_addr: 传感器 ip 地址。 失败: 其它

## 4.104 关闭力矩控制

函数	<b>disable_force_control ()</b>
描述	关闭力矩控制
参数	
返回值	成功: (0,) 失败: 其它

## 4.105 设置速度柔顺控制参数

函数	<b>set_vel_compliant_ctrl (level, rate1, rate2, rate3, rate4)</b>
描述	设置速度柔顺控制参数，速度柔顺控制有 3 个等级，有 4 个比率等级。
参数	level: 柔顺控制等级，等级分为：1, 2, 3 rate1: 比率等级 1 rate2: 比率等级 2 rate3: 比率等级 3 rate4: 比率等级 4 注： 比率等级之间的关系： $0 < \text{rate4} < \text{rate3} < \text{rate2} < \text{rate1} < 1$ ; (a) 当 level=1 时，只能设置 rate1, rate2, 而 rate3, rate4 的值全为 0 (b) 当 level=2 时，只能设置 rate1, rate2, rate3, 而 rate4 的值为 0 (c) 当 level=3 时，能设置 rate1, rate2, rate3, rate4 的值
返回值	成功：(0,) 失败：其它

## 4.106 设置柔顺控制力矩条件

函数	<b>set_compliance_condition (fx, fy, fz, tx, ty, tz)</b>
描述	设置柔顺控制力矩条件。
参数	fx: 沿着 x 轴受力，单位：N fy: 沿着 y 轴受力，单位：N fz: 沿着 z 轴受力，单位：N tx: 绕着 x 轴的扭矩，单位：Nm ty: 绕着 y 轴的扭矩，单位：Nm tz: 绕着 z 轴的扭矩，单位：Nm
返回值	成功：(0,) 失败：其它

## 4.107 设置网络异常时机器人自动终止运动类型

函数	<b>set_network_exception_handle (millisecond, mnt)</b>
描述	设置网络异常控制句柄，当网络出现异常情况时，控制机器人运动状态。
参数	millisecond: 时间参数，单位：ms。 mnt: 网络异常时机器人需要进行的动作类型, 0 代表机器人保持原来的运动，1 代表暂停运动，2 代表终止运动。
返回值	成功：(0,) 失败：其它

代码示例：

```
1. # -*- coding: utf-8 -*-
```

上海节卡机器人科技有限公司 Shanghai JAKA Robotics Ltd

电话 Tel : +400 006 2665 | 网站 Web:www.jaka.com

上海：上海市闵行区剑川路 610 号 33-35 幢 | Building 33-35, No.610 Jianchuan Rd, Minhang District, Shanghai

常州：江苏省常州市武进国家高新区武宜南路 377 号 10 号楼 | Building 10, No.377 South Wuyi Rd, Changzhou, Jiangsu

```

2. # import sys
3. # sys.path.append('D:\\vs2019ws\\PythonCtt\\PythonCtt')
4. import time
5. import jkrc
6. PI=3.1415926
7. #运动模式
8. ABS = 0
9. INCR= 1
10. robot = jkrc.RC("192.168.2.160")#返回机器人对象
11. robot.login()#登录
12. robot.power_on() #上电
13. robot.enable_robot()
14. robot.set_network_exception_handle(100,2)#设置 100ms，暂停运动。
15. print("move1")
16. num=0
17. while(1):
18.     robot.joint_move([1,1,1,1,1,1],ABS,False,0.5)
19.     robot.joint_move([-1,1,1,1,1,1],ABS,False,0.5)
20.     num = num +1
21.     print(num)
22.     time.sleep(6)
23. robot.logout()

```

## 4.108 设置机器人状态数据自动更新时间间隔

函数	set_network_exception_handle (millisecond)
描述	设置机器人状态数据自动更新时间间隔，可以改变系统占用率，默认尽可能快地发送以保证实时性
参数	millisecond: 时间参数，单位：ms。
返回值	成功：(0,) 失败：其它

代码示例：

```

1. import jkrc #导入模块
2. robot = jkrc.RC("192.168.2.165") #返回一个机器人对象
3. robot.login()
4. robot.set_status_data_update_time_interval(100)
5. robot.logout()

```

## 4.109 查询控制器 ftp 的目录

函数	<code>get_ftp_dir(remotedir, type);</code>
描述	查询控制器目录
参数	remotedir: 控制器内部文件夹名称 type: 0 文件和文件夹 1 文件 2 文件夹
返回值	成功: (0,ret)ret 为字符串 失败: 其它

代码示例:

```

1. import jkrc
2. robot = jkrc.RC("192.168.2.26")
3. robot.login()
4. dir= "/program/"
5. #登陆控制器, 需要将 192.168.2.26 替换为自己控制器的 IP
6. robot.init_ftp_client()
7. result = robot.get_ftp_dir("/program/", 0)
8. print(result)
9. robot.close_ftp_client()
10. robot.logout()

```

## 4.110 下载 FTP 文件

函数	<code>download_file(local, remote, opt)</code>
描述	从机器人控制柜的 ftp 下载文件或文件夹到本地, 查询轨迹“/track/”, 查询脚本程序“/program/”
参数	remote 控制器内部文件名绝对路径, 文件夹需要以“\”或“/”结尾取决于系统 如单个文件“/program/test/test.jks”或者文件夹“/program/test/” local 下载到本地文件名绝对路径 opt 1 单个文件 2 文件夹
返回值	成功: (0,) 失败: 其它

代码示例: 将 ftp 端的 program 文件夹下载到桌面 program 文件夹内

```

1. import jkrc
2. robot = jkrc.RC("192.168.2.26") #VMmodel
3. robot.login()
4. remote = "/program/"
5. local= "C:\\Users\\Administrator\\Desktop\\program\\track\\"
6. robot.init_ftp_client()
7. result = robot.download_file(local, remote, 2)
8. print(result)
9. robot.close_ftp_client()

```



```
10. robot.logout()
```

### 4.111 上传文件到 ftp

函数	<b>upload_file(local, remote, opt)</b>
描述	从控制器上传指定类型和名称的文件到本地
参数	remote 上传到控制器内部文件名绝对路径，文件夹需要以"\\"或"/"结尾取决于系统 local 本地文件名绝对路径 opt 1 单个文件 2 文件夹
返回值	成功: (0,) 失败: 其它

代码示例：将桌面的 lxxpro 文件夹内的所有文件和文件夹上传到 ftp 的 program/ 文件夹下

```
1. import jkrc
2. robot = jkrc.RC("192.168.2.26") #VMmodel
3. robot.login()
4. remote = "/program/"
5. local= "C:\\Users\\Administrator\\Desktop\\lxxpro\\"
6. robot.init_ftp_client()
7. result = robot.upload_file(local, remote, 2)
8. print(result)
9. robot.close_ftp_client()
10. robot.logout()
```

### 4.112 ftp 初始化

函数	<b>init_ftp_client()</b>
描述	初始化 ftp 客户端，与控制柜建立连接，可导入导出 program、track
参数	none
返回值	成功: (0,) 失败: 其它

### 4.113 ftp 关闭

函数	<b>close_ftp_client()</b>
描述	关闭 ftpclient
参数	none
返回值	成功: (0,) 失败: 其它

## 4.114 重命名 ftp 上的文件

函数	<code>upload_file(local, remote, opt)</code>
描述	重命名控制器指定类型和名称的文件
参数	remote 控制器内部文件名绝对路径，文件夹需要以“\”或“/”结尾取决于系统 des 重命名的目标名 opt 1 单个文件 2 文件夹，重命名文件时会重命名文件夹内所有文件，便于/track/使用
返回值	成功：(0,) 失败：其它

代码示例：将 ftp 的 lxxpro 文件夹内的所有文件和文件夹重命名为 lxx

```

1. import jkrc
2.
3. robot = jkrc.RC("192.168.2.26") #VMmodel
4. robot.login()
5. remote = "/lxxpro/"
6. des = "lxx"
7. robot.init_ftp_client()
8. result = robot.rename_ftp_file(remote, des, 2)
9. print(result)
10. robot.close_ftp_client()
11. robot.logout()

```

## 4.115 删除文件到 ftp

函数	<code>del_ftp_file( remote, opt)</code>
描述	从控制器上传指定类型和名称的文件到本地
参数	remote 控制器内部文件名绝对路径，文件夹需要以“\”或“/”结尾取决于系统 opt 1 单个文件 2 文件夹
返回值	成功：(0,) 失败：其它

代码示例：谨慎操作，该 demo 会删除所有程序！！！！

```

1. import jkrc
2.
3. robot = jkrc.RC("192.168.2.26") #VMmodel
4. robot.login()
5. dir= "/program/"
6. robot.init_ftp_client()
7. result = robot.del_ftp_file("/program/", 2)
8. print(result)

```

```
9. robot.close_ftp_client()  
10. robot.logout()
```

## 5. 反馈和勘误

文档中如若出现不准确的描述或者错误，恳请读者指正批评。如果您在阅读过程中发现任何问题或者有想提出的意见，可以发送邮件到 [support@jaka.com](mailto:support@jaka.com)，我们的同事会尽量一一回复。

## 产权说明

上海节卡机器人有限公司 版权所有。

上海节卡机器人有限公司对本文档中介绍的产品所包含的相关技术拥有知识产权。

本文档及相关产品按照限制其使用、复制、分发和反编译的许可证进行分发。未经上海节卡机器人有限公司事先书面授权，不得以任何方式、任何形式复制本产品或本文档的任何部分。

F