

UART串行总线舵机Python SDK使用手册

创建串行总线舵机管理器

串行总线舵机的文件路径 `fashionstar-uart-servo-python/src/userservo.py`，使用的时候可以将 `userservo.py` 拷贝至你当前工程文件夹里面。

或者使用的时候，将 `userservo.py` 所在的文件夹添加到系统路径里面，相对路径/绝对路径都可以。

```
1 # 添加userservo.py的系统路径
2 import sys
3 sys.path.append("../src")
```

然后使用的过程中一般需要导入如下这两个依赖

```
1 # PySerial 负责串口总线通信
2 import serial
3 # UartServoManager 串口总线舵机管理器
4 from userservo import UartServoManager
```

接下来要创建串口对象，指定相关的参数

```
1 # 参数配置
2 # 角度定义
3 SERVO_PORT_NAME = 'COM7' # 舵机串口号
4 SERVO_BAUDRATE = 115200 # 舵机的波特率
5 SERVO_ID = 0 # 舵机的ID号
6
7 # 初始化串口
8 uart = serial.Serial(port=SERVO_PORT_NAME, baudrate=SERVO_BAUDRATE,\
9                       parity=serial.PARITY_NONE, stopbits=1,\
10                      bytesize=8, timeout=0)
```

创建舵机管理器，将串口对象传入到构造器 `UartServoManager` 里面。

```
1 # 初始化舵机管理器
2 userservo = UartServoManager(uart)
```

舵机通信检测

API-ping

调用舵机的 `ping()` 函数用于舵机的通信检测, 判断舵机是否在线。

函数原型

```
1 def ping(self, servo_id:int):
```

输入参数

- `servo_id`: 舵机ID

输出参数

- `is_online`: 舵机是否在线

例程源码

example/ping.py

```
1  '''
2  FashionStar Uart舵机
3  > Python SDK舵机通讯检测 Example <
4  -----
5  * 作者: 深圳市华馨京科技有限公司
6  * 网站: https://fashionrobo.com/
7  * 更新时间: 2023/03/13
8  -----
9  '''
10 # 添加uservo.py的系统路径
11 import sys
12 sys.path.append(".././src")
13 # 导入依赖
14 import time
15 import serial
16 from uservo import UartServoManager
17
18 # 参数配置
19 # 角度定义
20 SERVO_PORT_NAME = 'COM7' # 舵机串口号
21 SERVO_BAUDRATE = 115200 # 舵机的波特率
22 SERVO_ID = 0 # 舵机的ID号
23
24 # 初始化串口
25 uart = serial.Serial(port=SERVO_PORT_NAME, baudrate=SERVO_BAUDRATE,\
26                       parity=serial.PARITY_NONE, stopbits=1,\
27                       bytesize=8,timeout=0)
28 # 初始化舵机管理器
29 uservo = UartServoManager(uart)
30
31 # 舵机通讯检测
32 is_online = uservo.ping(SERVO_ID)
33 print("舵机ID={} 是否在线: {}".format(SERVO_ID, is_online))
34
```



```
29 | uservo = UartServoManager(uart)
30 |
31 | power = 500 # 阻尼模式下的功率，单位mW
32 | uservo.set_damping(SERVO_ID, power)
```

舵机角度查询

API-query_servo_angle

函数原型

```
1 | def query_servo_angle(self, servo_id):
```

输入参数

- servo_id: 舵机ID

输出参数

- angle: 舵机角度(单圈/多圈)

注意事项

注意这里返回的角度是多圈模式的角度还是单圈模式的角度，取决于上次控制舵机的角度的指令是单圈模式还是/多圈模式，默认为单圈。

如果想人为的设定查询多圈/单圈，可以在查询之前设定 `uservo.servos[servo_id].is_mturn` 这个布尔值。

- is_mturn=True: 返回多圈角度
- is_mturn=False: 返回单圈角度

例程源码

设置舵机为阻尼模式，转动舵机 1s打印一下当前的角度。

example/query_servo_angle.py

```
1 | '''
2 | FashionStar Uart舵机
3 | > Python SDK舵机角度查询 Example <
4 | -----
5 | * 作者：深圳市华馨京科技有限公司
6 | * 网站：https://fashionrobo.com/
7 | * 更新时间：2023/03/13
8 | -----
9 | '''
10 | # 添加uservo.py的系统路径
11 | import sys
12 | sys.path.append(".././src")
13 | # 导入依赖
```

```

14 import time
15 import serial
16 from uservo import UartServoManager
17
18 # 参数配置
19 # 角度定义
20 SERVO_PORT_NAME = 'COM7' # 舵机串口号
21 SERVO_BAUDRATE = 115200 # 舵机的波特率
22 SERVO_ID = 0 # 舵机的ID号
23
24 # 初始化串口
25 uart = serial.Serial(port=SERVO_PORT_NAME, baudrate=SERVO_BAUDRATE,\
26                       parity=serial.PARITY_NONE, stopbits=1,\
27                       bytesize=8, timeout=0)
28 # 初始化舵机管理器
29 uservo = UartServoManager(uart)
30
31 # 设置舵机为阻尼模式
32 uservo.set_damping(SERVO_ID, 200)
33
34 # 舵机角度查询
35 while True:
36     angle = uservo.query_servo_angle(SERVO_ID)
37     print("当前舵机角度: {:.1f} °".format(angle), end='\r')
38     time.sleep(1)

```

设置舵机角度

多圈角度模式仅限于磁编码系列舵机，其他系列不兼容该模式

API-`set_servo_angle`

设置舵机角度，这个API包含了6种舵机角度控制模式，通过传入不同的参数调用不同的指令。具体的使用方式，可以参考[例程源码](#)。

函数原型

```

1 def set_servo_angle(self, servo_id:int, angle:float, is_mturn:bool=False,
    interval:float=None, velocity:float=None, t_acc:int=20, t_dec:int=20,
    power:int=0, mean_dps:float=100.0):

```

输入参数

- `servo_id`: 舵机的ID号
- `angle`: 目标角度
- `is_mturn`: 是否是多圈模式
- `interval`: 中间间隔，单位ms
- `velocity`: 舵机的目标转速，单位dps
- `t_acc`: 加速时间，启动时加速段的时间。单位ms
- `t_dec`: 减速时间，运动到接近目标的减速段时间。单位ms

- `power`: 功率限制, 单位mW
- `mean_dps`: 平均转速, 单位dps, 用于估计interval

输出参数

- 无

API-wait

等待所有的舵机到达目标角度。

函数原型

```
1 def wait(self, timeout=None):
```

输入参数

- `timeout`: 阻塞式等待的超时判断阈值, 单位ms

输出参数

- 无

例程源码

```
1  '''
2  FashionStar Uart舵机
3  > 设置舵机角度 <
4  -----
5  * 作者: 深圳市华馨京科技有限公司
6  * 网站: https://fashionrobo.com/
7  * 更新时间: 2023/03/13
8  -----
9  '''
10 # 添加uservo.py的系统路径
11 import sys
12 sys.path.append(".././src")
13 # 导入依赖
14 import time
15 import struct
16 import serial
17 from uservo import UartServoManager
18
19 # 参数配置
20 # 角度定义
21 SERVO_PORT_NAME = 'COM6' # 舵机串口号
22 SERVO_BAUDRATE = 115200 # 舵机的波特率
23 SERVO_ID = 0 # 舵机的ID号
24
25 # 初始化串口
26 uart = serial.Serial(port=SERVO_PORT_NAME, baudrate=SERVO_BAUDRATE, \
27                      parity=serial.PARITY_NONE, stopbits=1, \
28                      bytesize=8, timeout=0)
```

```

29 # 初始化舵机管理器
30 uservo = UartServoManager(uart, is_debug=True)
31
32 print("[单圈模式]设置舵机角度为90.0°")
33 uservo.set_servo_angle(SERVO_ID, 90.0, interval=0) # 设置舵机角度 极速模式
34 uservo.wait() # 等待舵机静止
35 print("-> {}".format(uservo.query_servo_angle(SERVO_ID)))
36
37 print("[单圈模式]设置舵机角度为-80.0°，周期1000ms")
38 uservo.set_servo_angle(SERVO_ID, -80.0, interval=1000) # 设置舵机角度(指定周期
    单位ms)
39 uservo.wait() # 等待舵机静止
40 print("-> {}".format(uservo.query_servo_angle(SERVO_ID)))
41
42 print("[单圈模式]设置舵机角度为70.0°，设置转速为200 °/s，加速时间100ms，减速时间
    100ms")
43 uservo.set_servo_angle(SERVO_ID, 70.0, velocity=200.0, t_acc=100, t_dec=100)
    # 设置舵机角度(指定转速 单位°/s)
44 uservo.wait() # 等待舵机静止
45 print("-> {}".format(uservo.query_servo_angle(SERVO_ID)))
46
47
48 print("[单圈模式]设置舵机角度为-90.0°，添加功率限制")
49 uservo.set_servo_angle(SERVO_ID, -90.0, power=400) # 设置舵机角度(指定功率 单位
    mW)
50 uservo.wait() # 等待舵机静止
51
52 #####
53
54 print("[多圈模式]设置舵机角度为900.0°，周期1000ms")
55 uservo.set_servo_angle(SERVO_ID, 900.0, interval=1000, is_mturn=True) # 设置
    舵机角度(指定周期 单位ms)
56 uservo.wait() # 等待舵机静止
57 print("-> {}".format(uservo.query_servo_angle(SERVO_ID)))
58
59 print("[多圈模式]设置舵机角度为-900.0°，设置转速为200 °/s")
60 uservo.set_servo_angle(SERVO_ID, -900.0, velocity=200.0, t_acc=100,
    t_dec=100, is_mturn=True) # 设置舵机角度(指定转速 单位°/s) dps: degree per
    second
61 uservo.wait() # 等待舵机静止
62 print("-> {}".format(uservo.query_servo_angle(SERVO_ID)))
63
64 print("[多圈模式]设置舵机角度为-850.0°，添加功率限制")
65 uservo.set_servo_angle(SERVO_ID, -850.0, power=400, is_mturn=True) # 设置舵机
    角度(指定功率 单位mW)
66 uservo.wait() # 等待舵机静止
67 print("-> {}".format(uservo.query_servo_angle(SERVO_ID)))
68

```

轮转模式

API-wheel_stop

轮转模式停止转动。

函数原型

```
1 def wheel_stop(self, servo_id):
```

输入参数

- `servo_id`: 舵机ID

输出参数

- 无

API-set_wheel_norm

设置轮转普通模式，转速单位: °/s

函数原型

```
1 def set_wheel_norm(self, servo_id, is_cw=True, mean_dps=None)
```

输入参数

- `servo_id`: 舵机ID
- `is_cw`: 是否是顺时针
 - `True`: 顺时针
 - `False`: 逆时针
- `mean_dps`: 平均转速

输出参数

- 无

API-set_wheel_turn

轮转模式，让舵机旋转特定的圈数。

函数原型

```
1 def set_wheel_turn(self, servo_id, turn=1, is_cw=True, mean_dps=None,  
    is_wait=True):
```

输入参数

- `servo_id`: 舵机ID
- `turn`: 目标要旋转的圈数
- `is_cw`: 旋转方向，是否为顺时针

- `True`: 顺时针
- `False`: 逆时针
- `mean_dps`: 平均转速
- `is_wait`: 是否是阻塞式等待

输出参数

- 无

API-`set_wheel_time`

轮转模式，旋转特定的时间。

函数原型

```
1 def set_wheel_time(self, servo_id, interval=1000, is_cw=True, mean_dps=None, is_wait=True):
```

输入参数

- `servo_id`: 舵机ID
- `interval`: 目标要旋转的时间, 单位ms
- `is_cw`: 旋转方向，是否为顺时针
 - `True`: 顺时针
 - `False`: 逆时针
- `mean_dps`: 平均转速, 单位dps
- `is_wait`: 是否是阻塞式等待

输出参数

- 无

例程源码

`src/wheel.py`

```
1 '''
2 FashionStar Uart舵机
3 > Python SDK 舵机轮转模式测试 <
4 -----
5 * 作者：深圳市华馨京科技有限公司
6 * 网站：https://fashionrobo.com/
7 * 更新时间：2023/03/13
8 -----
9 '''
10 # 添加uservo.py的系统路径
11 import sys
12 sys.path.append(".././src")
13 # 导入依赖
14 import time
```

```

15 import serial
16 from uservo import UartServoManager
17
18 # 参数配置
19 # 角度定义
20 SERVO_PORT_NAME = 'COM7' # 舵机串口号
21 SERVO_BAUDRATE = 115200 # 舵机的波特率
22 SERVO_ID = 0 # 舵机的ID号
23
24 # 初始化串口
25 uart = serial.Serial(port=SERVO_PORT_NAME, baudrate=SERVO_BAUDRATE,\
26                       parity=serial.PARITY_NONE, stopbits=1,\
27                       bytesize=8,timeout=0)
28 # 初始化舵机管理器
29 uservo = UartServoManager(uart)
30
31 print("测试常规模式")
32
33 # 设置舵机为轮转普通模式
34 # 旋转方向(is_cw) : 顺时针
35 # 角速度(mean_dps) : 单位°/s
36 uservo.set_wheel_norm(SERVO_ID, is_cw=True, mean_dps=200.0)
37
38 # 延时5s然后关闭
39 time.sleep(5.0)
40
41 # 停止
42 uservo.wheel_stop(SERVO_ID)
43
44 time.sleep(1)
45
46 # 定圈模式
47 print("测试定圈模式")
48 uservo.set_wheel_turn(SERVO_ID, turn=5, is_cw=False, mean_dps=200.0)
49
50 # 定时模式
51 print("测试定时模式")
52 uservo.set_wheel_time(SERVO_ID, interval=5000, is_cw=True, mean_dps=200.0)
53

```

用户配置表修改

API- reset_user_data

重置用户数据表, 恢复默认值。

函数原型

```
1 def reset_user_data(self, servo_id):
```

输入参数

- `servo_id`: 舵机ID

输出参数

- 无

API-`read_data`

读取数据。

函数原型

```
1 def read_data(self, servo_id, address):
```

输入参数

- `servo_id`: 舵机ID
- `address`: 内存表

输出参数

- `content`: 数值的二进制数据流

API-`write_data`

写入数据。

函数原型

```
1 def write_data(self, servo_id, address, content):
```

输入参数

- `servo_id`: 舵机ID
- `address`: 内存表
- `content`: 数值的二进制数据流

输出参数

- 无

例程源码-重置用户数据表

`example/reset_user_data.py`

```
1 '''
2 FashionStar Uart舵机
3 > 内存表数据重置 <
4
5 注意事项：重置内存表这个指令比较特殊，舵机ID也会被重置为0
6 因此测试该指令的时候，最好只接一颗舵机。
```

```
7 -----
8 * 作者：深圳市华馨京科技有限公司
9 * 网站：https://fashionrobo.com/
10 * 更新时间：2023/03/13
11 -----
12 '''
13 # 添加uservo.py的系统路径
14 import sys
15 sys.path.append(".././src")
16 # 导入依赖
17 import time
18 import struct
19 import serial
20 from uservo import UartServoManager
21
22 # 参数配置
23 # 角度定义
24 SERVO_PORT_NAME = 'COM7' # 舵机串口号
25 SERVO_BAUDRATE = 115200 # 舵机的波特率
26 SERVO_ID = 0 # 舵机的ID号
27
28 # 数据表定义
29 ADDRESS_SOFTSTART = 49 # 上电缓启动地址位
30 SOFTSTART_OPEN = 1 # 上电缓启动-开启
31 SOFTSTART_CLOSE = 0 # 上电缓启动-关闭
32
33 # 初始化串口
34 uart = serial.Serial(port=SERVO_PORT_NAME, baudrate=SERVO_BAUDRATE,\
35                       parity=serial.PARITY_NONE, stopbits=1,\
36                       bytesize=8,timeout=0)
37 # 初始化舵机管理器
38 uservo = UartServoManager(uart, is_debug=True)
39 # 重置用户数据
40 uservo.reset_user_data(SERVO_ID)
41
42
43 # 舵机扫描
44 print("开始进行舵机扫描")
45 uservo.scan_servo()
46 servo_list = list(uservo.servos.keys())
47 print("舵机扫描结束，舵机列表：{}".format(servo_list))
48
49 if SERVO_ID not in servo_list:
50     print("指定的SERVO_ID无效，请修改舵机ID列表")
51     exit(-1)
52
53 print("重置舵机内存表：舵机ID = {}".format(SERVO_ID))
54 uservo.reset_user_data(SERVO_ID)
55
56 print("重新进行舵机扫描")
57 uservo.scan_servo()
58 servo_list = list(uservo.servos.keys())
59 print("舵机扫描结束，舵机列表：{}".format(servo_list))
60
```

例程源码-读取内存表

example/read_data.py

```
1  '''
2  FashionStar Uart舵机
3  > 内存表数据读取 <
4  -----
5  * 作者：深圳市华馨京科技有限公司
6  * 网站：https://fashionrobo.com/
7  * 更新时间：2023/03/13
8  -----
9  '''
10 # 添加uservo.py的系统路径
11 import sys
12 sys.path.append(".././src")
13 # 导入依赖
14 import time
15 import struct
16 import serial
17 from uservo import UartServoManager
18
19 # 参数配置
20 # 角度定义
21 SERVO_PORT_NAME = 'COM7' # 舵机串口号
22 SERVO_BAUDRATE = 115200 # 舵机的波特率
23 SERVO_ID = 0 # 舵机的ID号
24 # 数据表定义
25 ADDRESS_VOLTAGE = 1 # 总线电压值的地址
26
27 # 初始化串口
28 uart = serial.Serial(port=SERVO_PORT_NAME, baudrate=SERVO_BAUDRATE, \
29                       parity=serial.PARITY_NONE, stopbits=1, \
30                       bytesize=8, timeout=0)
31 # 初始化舵机管理器
32 uservo = UartServoManager(uart)
33
34 # 内存表读取
35 # 注：因为每个数据位数据格式各不相同
36 # 因此读取得到的是字节流
37 voltage_bytes = uservo.read_data(SERVO_ID, ADDRESS_VOLTAGE)
38 # 数据解析
39 # 电压的数据格式为uint16_t,单位：mV
40 # 关于struct的用法，请参阅官方手册：
41 # https://docs.python.org/3/library/struct.html
42 voltage = struct.unpack('<H', voltage_bytes)
43 print("总线电压 {} mV".format(voltage))
```

例程源码-写入内存表

example/write_data.py

```
1  '''
2  FashionStar Uart舵机
3  > 内存表数据写入 <
4  -----
5  * 作者：深圳市华馨京科技有限公司
6  * 网站：https://fashionrobo.com/
7  * 更新时间：2023/03/13
8  -----
9  '''
10 # 添加uservo.py的系统路径
11 import sys
12 sys.path.append(".././src")
13 # 导入依赖
14 import time
15 import struct
16 import serial
17 from uservo import UartServoManager
18
19 # 参数配置
20 # 角度定义
21 SERVO_PORT_NAME = 'COM7' # 舵机串口号
22 SERVO_BAUDRATE = 115200 # 舵机的波特率
23 SERVO_ID = 0 # 舵机的ID号
24
25 # 数据表定义
26 ADDRESS_SOFTSTART = 49 # 上电缓启动地址位
27 SOFTSTART_OPEN = 1 # 上电缓启动-开启
28 SOFTSTART_CLOSE = 0 # 上电缓启动-关闭
29
30 # 初始化串口
31 uart = serial.Serial(port=SERVO_PORT_NAME, baudrate=SERVO_BAUDRATE,\
32                       parity=serial.PARITY_NONE, stopbits=1,\
33                       bytesize=8,timeout=0)
34 # 初始化舵机管理器
35 uservo = UartServoManager(uart)
36
37 # 内存表写入
38 # 注：在写入之前，需要查阅手册确保该数据位可写
39 # 缓启动数据类型 uint8_t, 首先构造数据位
40 softstart_bytes = struct.pack('<B', SOFTSTART_OPEN)
41 # 将数据写入内存表
42 ret = uservo.write_data(SERVO_ID, ADDRESS_SOFTSTART, softstart_bytes)
43 # 打印日志
44 print("缓启动数据写入是否成功: {}".format(ret))
```

系统状态查询

API-query_voltage

查询当前的电压

函数原型

```
1 def query_voltage(self, servo_id)
```

输入参数

- servo_id: 舵机ID

输出参数

- voltage: 电压, 单位V

API-query_current

查询当前的电流

函数原型

```
1 def query_current(self, servo_id):
```

输入参数

- servo_id: 舵机ID

输出参数

- power: 舵机电流, 单位A

API-query_power

查询当前的功率

函数原型

```
1 def query_power(self, servo_id)
```

输入参数

- servo_id: 舵机ID

输出参数

- power: 舵机功率, 单位W

API-query_temperature

查询舵机当前的温度

函数原型

```
1 def query_temperature(self, servo_id)
```

输入参数

- `servo_id`: 舵机ID

输出参数

- `temperature`: 温度, ADC值

例程源码

example/servo_status.py

```
1  '''
2  FashionStar Uart舵机
3  > 读取舵机的状态信息 <
4  -----
5  * 作者: 深圳市华馨京科技有限公司
6  * 网站: https://fashionrobo.com/
7  * 更新时间: 2023/03/13
8  -----
9  '''
10 # 添加uservo.py的系统路径
11 import sys
12 sys.path.append("../src")
13 # 导入依赖
14 import time
15 import serial
16 from uservo import UartServoManager
17
18 # 参数配置
19 # 角度定义
20 SERVO_PORT_NAME = 'COM7' # 舵机串口号
21 SERVO_BAUDRATE = 115200 # 舵机的波特率
22 SERVO_ID = 0 # 舵机的ID号
23
24 # 初始化串口
25 uart = serial.Serial(port=SERVO_PORT_NAME, baudrate=SERVO_BAUDRATE,\
26                      parity=serial.PARITY_NONE, stopbits=1,\
27                      bytesize=8, timeout=0)
28 # 初始化舵机管理器
29 uservo = UartServoManager(uart)
30
31 def log_servo_status():
32     '''打印舵机状态'''
33     # 读取温度
34     voltage = uservo.query_voltage(SERVO_ID)
```



```
35 # 读取电流
36 current = uservo.query_current(SERVO_ID)
37 # 读取功率
38 power = uservo.query_power(SERVO_ID)
39 # 读取温度
40 temp = uservo.query_temperature(SERVO_ID)
41
42 print("voltage: {:.4f}V; Current: {:.4f}A; Power: {:.4f}W; T:
{:2.0f}").format(\
43     voltage, current, power, temp), end='\r')
44
45 while True:
46     uservo.set_servo_angle(SERVO_ID, 90)
47     while not uservo.is_stop():
48         log_servo_status()
49         time.sleep(0.1)
50
51     time.sleep(1)
52
53     uservo.set_servo_angle(SERVO_ID, -90)
54     while not uservo.is_stop():
55         log_servo_status()
56         time.sleep(0.1)
57
58     time.sleep(1)
```

注: ADC转换为摄氏度值公式

$R1 = 10000;$ (NTC 分压电阻值, $R1$ 靠近电源, NTC 靠近 GND)

$Rt25 = 10000;$ (NTC 在 25° 室温时的电阻值)

$B = 3435;$ (NTC 的材料常数)

$K = 273.15;$ (绝对零度)

$T = 1/(\ln(R1 \cdot ADC \div Rt25 \div (4096 - ADC)) \div B + 1 \div (K + 25)) - K$

温度(°C)	ADC	温度(°C)	ADC	温度(°C)	ADC
50	1191	60	941	70	741
51	1164	61	918	71	723
52	1137	62	897	72	706
53	1110	63	876	73	689
54	1085	64	855	74	673
55	1059	65	835	75	657
56	1034	66	815	76	642
57	1010	67	796	77	627
58	986	68	777	78	612

温度(℃)	ADC	温度(℃)	ADC	温度(℃)	ADC
59	963	69	759	79	598

- 以上为50-79℃ 温度/ADC参照表