**Yanshee Robot SDK API**

**使用说明**

**V1.0 (2018/04/08)**

目录

[1. 文档概述 1](#_Toc522283735)

[2. 准备工作 1](#_Toc522283736)

[3. API接口说明 1](#_Toc522283737)

[3.1软件版本 1](#_Toc522283738)

[3.2读取机器状态 2](#_Toc522283739)

[3.3舵机控制 2](#_Toc522283740)

[3.4舵机回读 3](#_Toc522283741)

[3.5音量调节 3](#_Toc522283742)

[3.6局部动作控制 3](#_Toc522283743)

[3.7读取传感器值 4](#_Toc522283744)

[3.8控制灯光 4](#_Toc522283745)

[3.9按键事件检测 4](#_Toc522283746)

[3.10播放动作文件 5](#_Toc522283747)

[3.11动作暂停 5](#_Toc522283748)

[3.12声音播放 5](#_Toc522283749)

[3.13 TTS播放 5](#_Toc522283750)

[3.14透传指令接口 6](#_Toc522283751)

[3.15检测手机状态 6](#_Toc522283752)

[3.16检测是否听到xxx 6](#_Toc522283753)

[3.17拍照 7](#_Toc522283754)

[3.18上报状态 7](#_Toc522283755)

[3.19视觉检测接口 7](#_Toc522283756)

[4. 调用举例 8](#_Toc522283757)

[4.1 Python调用举例 8](#_Toc522283758)

[4.2执行python文件 12](#_Toc522283759)

[4.3注意事项 12](#_Toc522283760)

## 文档概述

本文档是开发优必选机器人(Yanshee)应用程序的用户使用指南。第三方开发者可以通过接入SDK来控制机器人的各舵机转动和读传感器值或者让机器人说话，还集成人脸识别和手势识别等功能。本文主要描述SDK的使用方法，用于指导开发人员快速进行基于Python/JAVA/JS等语言应用的二次开发。

## 准备工作

给Yanshee机器人安装最新版（V1.2.0以上）的软件包,并下载最新版SDK版本。详见GitHub说明链接：<https://github.com/UBTEDU/yanshee-raspi-sdk>

机器人已经默认安装了SDK依赖文件，可以直接把example目录下的python文件拷贝到任意机器人目录（例如/home/pi）下面直接运行。

## API接口说明

### 3.1软件版本

|  |  |
| --- | --- |
| 名称 | bool ubtGetSWVersion(int Type, char\*Version, int VersionLen) |
| 描述 | 获取机器的硬件版本信息 |
| 参数 | 【Type】  0 为STM32 app版本；1~20为舵机版本；30：为SDK版本；31：为Raspberry APP版本  【Version】对应的版本信息字符串，最长为20Byte  【VersionLen】版本字符串长度，最大为20Byte |
|  |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

### 3.2读取机器状态

|  |  |
| --- | --- |
| 名称 | bool ubtGetRobotStatus(int Type, char\* Status) |
| 描述 | 获取机器的状态信息 |
| 参数 | 【Type】  1 为播放状态（Status：2 暂停， 1 播放，0空闲）；  2 为音量（Status：0~ff）；  3 为电量（Status：  [0]是否充电（0否，1是），  [1-2]电压值(mv，十六进制，高位在前)，  [3]电量百分比（取值范围0~100，十六进制）； |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

### 3.3舵机控制

|  |  |
| --- | --- |
| 名称 | int ubtSetRobotServo(UBTEDU\_ROBOTSERVO\_T \*servoAngle, int iTime); |
| 描述 | 控制 单个、多个、所有 舵机的运动 |
| 参数 | 【servoAngle】包含了1到17号舵机的0 – 180度对应角度。  【Time】舵机运行时间，单位为20ms（最小大于10） |
| 返回值 | -1为失败， 0为成功 |
| 调用方式 | 同步 |

举例：此接口举例向前请参考第四节例3部分。

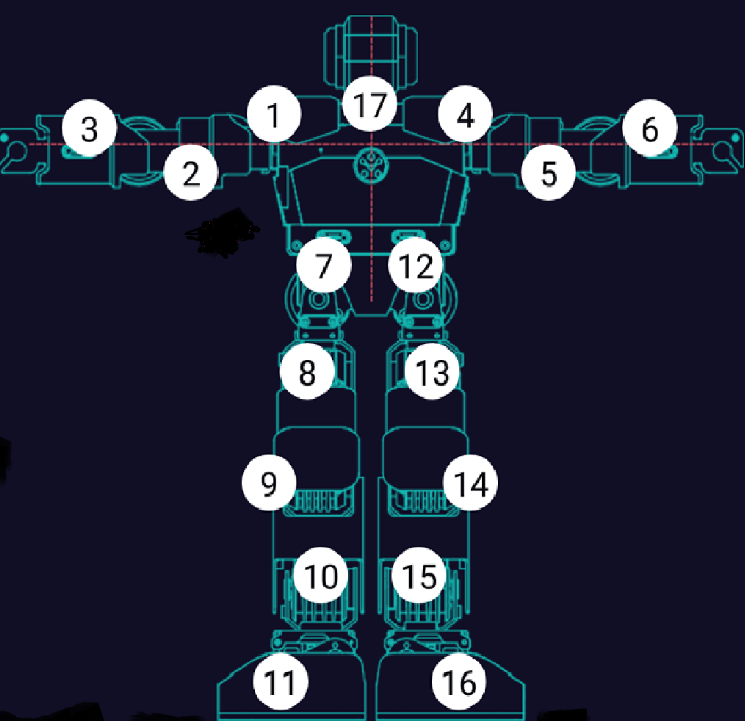


图1 Yanshee机器人正面舵机分布说明图

### 3.4舵机回读

|  |  |
| --- | --- |
| 名称 | void ubtGetRobotServo(UBTEDU\_ROBOTSERVO\_T \*servoAngle) |
| 描述 | 回读 单个、多个、所有舵机的角度，直接返回全部舵机角度值，通过类似下面结构来读取：  servoinfo = RobotApi.UBTEDU\_ROBOTSERVO\_T()  servoinfo.SERVO1\_ANGLE  servoinfo.SERVO2\_ANGLE  ...........................  servoinfo.SERVO17\_ANGLE |
| 参数 | 【servoAngle】 0 - 180 对应舵机角度，值为255表示舵机无应答 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

举例：详情请参考第四部分第4个例子：读取舵机举例。

### 3.5音量调节

|  |  |
| --- | --- |
| 名称 | bool ubtSetRobotVolume(int Volume) |
| 描述 | 设置音量 |
| 参数 | 【Volume】音量取值范围0~255 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

### 3.6局部动作控制

|  |  |
| --- | --- |
| 名称 | bool ubtSetRobotMotion(char\* Type, char\* direct, int speed, int repeat) |
| 描述 | 控制机器人做局部动作，例如：蹲下、举手、 |
| 参数 | 【Type】：crouch蹲下、raise 举手、 stretch 伸手、come on招手、wave挥手、bend 弯曲、walk 走路、turn around 转身、head 转动头部、bow鞠躬；  【direct】：left 左、right右、both双手、front前、back后;  【speed】：1/2/3/4/5（五个速度等级，依次更快），默认为3；  【Repeat】重复次数； |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

### 3.7读取传感器值

|  |  |
| --- | --- |
| 名称 | int ubtReadSensorValue( char \*Type, void \*Value, int ValueLen) |
| 描述 | 读传感器值（默认读取一个低位地址） |
| 参数 | 【Type】 gryo 陀螺仪、environment环境、board板上、infrared红外、 ultrasonic超声波、touch触摸、color颜色、pressure压力、gas气体  【Value】返回传感器的值  【Valuelen】返回值最大长度 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

|  |  |
| --- | --- |
| 名称 | int ubtReadSensorValueByAddr (char \*Type, int Addr, void \*Value, int ValueLen ) |
| 描述 | 读对应地址的传感器值 |
| 参数 | 【Type】 gryo 陀螺仪、environment环境、board板上、infrared红外、 ultrasonic超声波、touch触摸、color颜色、pressure压力、gas气体  【Addr】需要读取模块的传感器IIC地址  【Value】返回传感器的值  【Valuelen】返回值最大长度 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

### 3.8控制灯光

|  |  |
| --- | --- |
| 名称 | bool ubtSetRobotLED( char \*Type, char \*Color, char \*Mode) |
| 描述 | 设置 机器人LED灯 |
| 参数 | 【Type】button camera mic  【color】灯颜色 white-白 red-红 green-绿 blue-蓝 yellow-黄 purple-紫 cyan-青绿；  【mode】off-关on-常亮 flicker-频闪 breath-呼吸 alternation 轮流 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

### 3.9按键事件检测

|  |  |
| --- | --- |
| 名称 | int ubtEventDetect(char \*pcEventType, char \*pcValue, int iTimeout) |
| 描述 | 执行动作文件 |
| 参数 | 【 pcEventType】要检测事件类型，目前默认只支持button类型；  【 pcValue】事件重复次数，0：没有发生 其它是实际发生次数；  【 iTimeout】设置事件超时时间，最小10s,最大10min。 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 异步 |

### 3.10播放动作文件

|  |  |
| --- | --- |
| 名称 | bool ubtStartRobotAction (char\* Name, int repeat) |
| 描述 | 执行动作文件 |
| 参数 | 【Name】动作文件名，例如：push up（俯卧撑）、bow（鞠躬）等  【Repeat】重复次数 0：连续执行 其它是实际次数 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 异步 |

### 3.11动作暂停

|  |  |
| --- | --- |
| 名称 | bool ubtStopRobotAction (void) |
| 描述 | 停止执行动作文件 |
| 参数 | 空 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

### 3.12声音播放

|  |  |
| --- | --- |
| 名称 | bool ubtPlayMusic(char\*Path, char\* Name) |
| 描述 | 执行声音文件 |
| 参数 | 【Path】动作文件所在目录，可以为default；  【Name】声音文件名； |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 异步 |

### 3.13 TTS播放

|  |  |
| --- | --- |
| 名称 | bool ubtVoiceTTS ( int isInterrputed, char \*tts) |
| 描述 | 将文字转成声音播放 |
| 参数 | 【isInterrputed】0 不打断前面的语句 1 打断前面的语句  【tts】文字信息 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 异步 |

### 3.14透传指令接口

|  |  |
| --- | --- |
| 名称 | bool ubtTransmitCMD(char\*Transmit, char\* Receive) |
| 描述 | 透传指令到舵机总线上 |
| 参数 | 【Transmit】发送到舵机总线上的内容；  【Receive】收到的反馈，只能收到固定20Bytes,格式参加舵机协议的起始和结束；  注意：传输的数据内容为数据的字符形式，比如：要发送 0xfe 0x01 0x0d,则Transmit为 ”FE010D”；Receive反之。 |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

### 3.15检测手机状态

|  |  |
| --- | --- |
| 名称 | int ubtCheckAPPStatus(char\* status, int timeout) |
| 描述 | 检测手机的状态 |
| 参数 | 【status】 static 静止、slant\_forward 前倾斜、slant\_backward后倾斜、slant\_left左倾斜、slant\_right右倾斜、forward\_and\_back前后晃动、swaying左右晃动  【timeout】检测超时时间，最小10s,最大10min; |
| 返回值 | -1获取状态超时  0检测ok |
| 调用方式 | 同步 |

### 3.16检测是否听到xxx

|  |  |
| --- | --- |
| 名称 | int ubtDetectVoiceMsg(char\* buf, int timeout) |
| 描述 | 检测是否听到 |
| 参数 | 【buf】 文字内容（utf-8编码）  【timeout】检测超时时间，最小10s,最大10min; |
| 返回值 | -1获取状态超时、  0检测ok |
| 调用方式 | 同步 |

### 3.17拍照

|  |  |
| --- | --- |
| 名称 | int ubtTakeAPhoto(char\* Name， int NameLen) |
| 描述 | 拍一张图片 |
| 参数 | 【Name】 图标名字，不带后缀（.jpg），如果为空则按照：img\_yyyymmdd\_hhmmss.jpg  【NameLen】图片名称长度 |
| 返回值 | -1获取状态超时  0检测ok |
| 调用方式 | 同步 |

### 3.18上报状态

|  |  |
| --- | --- |
| 名称 | int ubtReportStatusToApp(char\* name, char \*string) |
| 描述 | 上报状态到APP |
| 参数 | 【name】系统参数  【string】向手机上报的状态，目前有：start、run、finish |
| 返回值 | -1成功  0失败 |
| 调用方式 | 同步 |

### 3.19视觉检测接口

|  |  |
| --- | --- |
| 名称 | int ubtVisionDetect ( char \*Type, char \*Value, int Timeout) |
| 描述 | 检测是否看到xxx |
| 参数 | 【type】检测类型，取值：face 人脸，hand 手;  【value】返回检测到的人脸个数或者人手个数  【timeout】检测超时时间，最小10s,最大10min; |
| 返回值 | -1获取状态超时、  0检测ok |
| 调用方式 | 同步 |

## 调用举例

### 4.1 Python调用举例

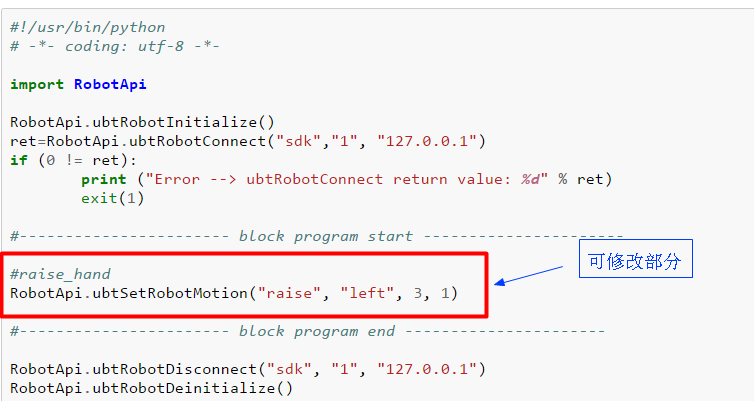
**Yanshee SDK代码编辑流程概述：**



**基本说明：**下面的例子前三个属于连接本地机器人并实现相应的动作控制的例子。如果只需要控制自身的机器人就可以直接按照例1中的说明修改框框部分即可。如果需要远程控制另外一台机器人，就必须按照例4的格式，修改目标机器人名称（本例中为Yanshee\_8F83）然后实现远程连接并控制远程机器人的目的。

**例1：**这个例子通过调用我们在第三部分讲的API的第六个接口（局部动作控制）来实现一个简单的控制Yanshee机器人的动作。

我们开发时候只需要修改虚线部分以内的代码即可。（如下图中红色方框部分）



这句话的意思是让机器人举左手，速度为3，重复1次。请参考第三部分的第6个API接口说明。

|  |  |
| --- | --- |
| 名称 | bool ubtSetRobotMotion(char\* Type, char\* direct, int speed, int repeat) |
| 描述 | 控制机器人做局部动作，例如：蹲下、举手、 |
| 参数 | 【Type】：crouch蹲下、raise 举手、 stretch 伸手、come on招手、wave挥手、bend 弯曲、walk 走路、turn around 转身、head 转动头部、bow鞠躬；  【direct】：left 左、right右、both双手、front前、back后;  【speed】：1/2/3/4/5（五个速度等级，依次更快），默认为3；  【Repeat】重复次数； |
| 返回值 | -1-- 操作失败；0 -- 操作成功 |
| 调用方式 | 同步 |

这里我们的Type填写的是raise，direct填的是left，speed写的3，Repeat写的1。

这样就实现了一个简单的机器人举左手一次的动作。

**例2：**这个例子通过设置舵机角度API实现了舵机2为60度和舵机3为40度的过程：

*#!/usr/bin/python*

*# -\*- coding: utf-8 -\*-*

**import** **RobotApi**

RobotApi.ubtRobotInitialize()

ret=RobotApi.ubtRobotConnect("sdk","1", "127.0.0.1")

**if** (0 != ret):

**print** ("Error --> ubtRobotConnect return value: ***%d***" % ret)

exit(1)

*#----------------------- block program start ----------------------*

servoinfo = RobotApi.UBTEDU\_ROBOTSERVO\_T()

*#set servo2 = 60°,servo3 = 40°*

servoinfo.SERVO2\_ANGLE = 60

servoinfo.SERVO3\_ANGLE = 40

ret = RobotApi.ubtSetRobotServo(servoinfo, 20)

*#----------------------- block program end ----------------------*

RobotApi.ubtRobotDisconnect("sdk", "1", "127.0.0.1")

RobotApi.ubtRobotDeinitialize()

**例3：**通过这个例子实现了对本地机器人所有舵机的读取：

*#!/usr/bin/python*

*# -\*- coding: utf-8 -\*-*

**import** **RobotApi**

RobotApi.ubtRobotInitialize()

ret=RobotApi.ubtRobotConnect("sdk","1", "127.0.0.1")

**if** (0 != ret):

**print** ("Error --> ubtRobotConnect return value: ***%d***" % ret)

exit(1)

*#----------------------- block program start ----------------------*

servoinfo = RobotApi.UBTEDU\_ROBOTSERVO\_T()

ret = RobotApi.ubtGetRobotServo(servoinfo)

**print** "servoinfo.SERVO1\_ANGLE = ***%d***" %(servoinfo.SERVO1\_ANGLE)

**print** "servoinfo.SERVO2\_ANGLE = ***%d***" %(servoinfo.SERVO2\_ANGLE)

**print** "servoinfo.SERVO3\_ANGLE = ***%d***" %(servoinfo.SERVO3\_ANGLE)

**print** "servoinfo.SERVO17\_ANGLE = ***%d***" %(servoinfo.SERVO17\_ANGLE)

*#----------------------- block program end ----------------------*

RobotApi.ubtRobotDisconnect("sdk", "1", "127.0.0.1")

RobotApi.ubtRobotDeinitialize()

**例4：**远程控制举例：这个例子通过SDK接口读取指定机器人(Yanshee\_8F83)的红外距离传感器数据，并将它打印出来。

*#!/usr/bin/python*

*# \_\*\_ coding: utf-8 -\*-*

**import** **time**

**import** **RobotApi**

RobotApi.ubtRobotInitialize()

*#------------------------------Connect------------------------------*

gIPAddr = ""

robotinfo = RobotApi.UBTEDU\_ROBOTINFO\_T()

*#The robot name you want to connect*

robotinfo.acName="Yanshee\_8F83"

ret = RobotApi.ubtRobotDiscovery("SDK", 15, robotinfo)

**if** (0 != ret):

**print** ("Return value: ***%d***" % ret)

exit(1)

gIPAddr = robotinfo.acIPAddr

ret = RobotApi.ubtRobotConnect("SDK", "1", gIPAddr)

**if** (0 != ret):

**print** ("Can not connect to robot ***%s***" % robotinfo.acName)

exit(1)

*#---------------------------Read Sensor Value------------------------*

infrared\_sensor = RobotApi.UBTEDU\_ROBOTINFRARED\_SENSOR\_T()

ret = RobotApi.ubtReadSensorValue("infrared",infrared\_sensor,4)

**if** ret != 0:

**print**("Can not read Sensor value. Error code: ***%d***" % (ret))

**else**:

**print**("Read Sensor Value: ***%d***" % (infrared\_sensor.iValue))

*#---------------------------Disconnect------------------------------*

RobotApi.ubtRobotDisconnect("SDK","1",gIPAddr)

RobotApi.ubtRobotDeinitialize()

### 4.2执行python文件

将这个例1的python文件保存为SDKSampleAction1.py（当然这个文件名可以按照您的喜好任意命名）放到机器人的树莓派目录/home/pi下面，然后执行此文件。

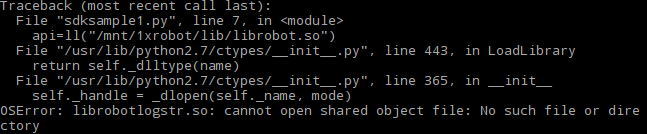
在命令行敲入以下命令

sudo python SDKSampleAction1.py

然后就会看到Yanshee机器人做的相应的动作。如果您想让机器人做其它动作、说话、或者读取传感器数据等，只要您能想到的场景都可以试试调用相应的python接口，完成相应的功能。

### 4.3注意事项

注意如果您在执行过程中遇到这个问题：



请先执行下面命令：

export LD\_LIBRARY\_PATH=$LB\_LIBRARY\_PATH:/mnt/1xrobot/lib

然后再执行

python SDKSampleAction1.py

非常感谢您的阅读！

另外附上例1源代码（可直接复制粘贴）：

*#!/usr/bin/python*

*# -\*- coding: utf-8 -\*-*

**import** **RobotApi**

RobotApi.ubtRobotInitialize()

ret=RobotApi.ubtRobotConnect("sdk","1", "127.0.0.1")

**if** (0 != ret):

**print** ("Error --> ubtRobotConnect return value: ***%d***" % ret)

exit(1)

*#----------------------- block program start ----------------------*

*#raise\_hand*

RobotApi.ubtSetRobotMotion("raise", "left", 3, 1)

*#----------------------- block program end ----------------------*

RobotApi.ubtRobotDisconnect("sdk", "1", "127.0.0.1")

RobotApi.ubtRobotDeinitialize()

以下无正文。