pyhula

hula python编程依赖包。
A Python pack used by hula.

```
python版本:3.6.7
```

安装 / Installing

在终端执行如下指令以安装pyhula,两种方式安装。
Input the following code in powershell(cmd.exe) to install pyhula.

```
pip install pyhula

pip install pyhula-1.0.4-cp36-cp36m-win_amd64.whl
```

查看版本 / Checking version

- 在控制终端执行pip list 进行查看
  Input "pip list" in powershell(cmd.exe) to get pyhula's verison
- 在程序中执行
  Using the following code.

```
import pyhula
ver = pyhula.get_version()
print(ver)
```

使用 / Usage

使用以下代码获取一个UserApi实例后，可以通过UserApi所提供的接口对hula无人机进行控制。接口说明请查看doc/html/中文/index.html文件。
Use the following codes to create a userApi instance. Its interfaces can be used to control fylo plane. Go to doc/html/English/index.html to see the interface specification.

```
import pyhula
api = pyhula.UserApi()
if not api.connect():
    print("connect error")
else:
    print('connection to  station by wifi')
```

```
api.single_fly_takeoff()#起飞
api.single_fly_touchdown()  #降落
```

接口说明 / Interface

连接无人机

```
connect(server_ip)

'''
    描述:
        连接无人机,
    参数:
        选填:server_ip: 无人机IPv4地址 不填自动获取
    返回值:
        True:成功 False:失败
'''
示例: api.connect('192.168.1.118')
示例: api.connect()
```

起飞

```
single_fly_takeoff(led)
'''
    描述:
    实时控制无人机起飞
    参数:led不填默认为0, 格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域, mode: 1/常
亮, 2/灭灯, 4/RGB 三色循环, 16/七彩灯,32/闪烁,64/呼吸灯
'''
示例: api.single_fly_takeoff()
      api.single_fly_takeoff({'r':16,'g':15,'b':100,'mode':1})
```

降落

```
single_fly_touchdown(led)
'''
    描述:
    实时控制无人机降落
    参数:led不填默认为0, 格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域, mode: 1/常
亮, 2/灭灯, 4/RGB 三色循环, 16/七彩灯,32/闪烁,64/呼吸灯
'''
示例:api.single_fly_touchdown()
    api.single_fly_touchdown({'r':16,'g':15,'b':100,'mode':1})
```

悬停

```
single_fly_hover_flight(time,led)
'''
    描述:
    飞机悬停
    参数:
    time:悬停时间（秒）
'''
示例: api.single_fly_hover_flight(10)
     api.single_fly_hover_flight(10,{'r':16,'g':15,'b':100,'mode':1})
```

向前飞

```
single_fly_forward(distance,speed,led)
'''
    描述:
    实时控制无人机向前飞
    参数:
    distance:飞行距离（厘米）
    speed:不填默认为100 速度（0-100cm）/s
    led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode：1/常亮，
2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
示例: api.single_fly_forward(100)
     api.single_fly_forward(100,100,{'r':16,'g':15,'b':100,'mode':1})
```

向后飞

```
single_fly_back(distance,speed,led)
'''
    描述:
        实时控制无人机向后飞
    参数:
        distance:飞行距离（厘米）
        speed:不填默认为100 速度（0-100cm）/s
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode：1/常
亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
示例: api.single_fly_back(100)
     api.single_fly_back(100,100,{'r':16,'g':15,'b':100,'mode':1})
```

向左飞

```
single_fly_left(distance,speed,led)
'''
    描述:
        实时控制无人机向左飞
    参数:
        distance:飞行距离（厘米）
        speed:不填默认为100 速度（0-100cm）/s
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode：1/常亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
示例：api.single_fly_left(100)
     api.single_fly_left(100,100,{'r':16,'g':15,'b':100,'mode':1})
```

向右飞

```
single_fly_right(distance,speed,led)
'''
    描述:
        实时控制无人机向右飞
    参数:
        distance:飞行距离（厘米）
        speed:不填默认为100 速度（0-100cm）/s
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode：1/常亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
        '''
示例：api.single_fly_right(100)
     api.single_fly_right(100,100,{'r':16,'g':15,'b':100,'mode':1})
```

向上飞

```
single_fly_up(distance,speed,led)
'''
    描述:
        实时控制无人机向上飞
    参数:
        height:飞行高度（厘米）
        speed:不填默认为100 速度（0-100cm）/s
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode：1/常亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
        '''
示例：api.single_fly_up(100)
     api.single_fly_up(100,100,{'r':16,'g':15,'b':100,'mode':1})
```

向下飞

```
single_fly_down(distance,speed,led)
'''
    描述:
        实时控制无人机向下飞
    参数:
        height:飞行高度（厘米）
        speed:不填默认为100 速度（0-100cm）/s
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode: 1/常亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
示例: api.single_fly_down(100)
     api.single_fly_down(100,100,{'r':16,'g':15,'b':100,'mode':1})
```

左旋转

```
single_fly_turnleft(angle,led)
'''
    描述:
        实时控制无人机向左转
    参数:
        angle:旋转角度（度）
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode: 1/常亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
示例: api.single_fly_turnleft(90)
     api.single_fly_turnleft(90,{'r':16,'g':15,'b':100,'mode':1})
```

右旋转

```
single_fly_turnright(angle,led)
'''
    描述:
        实时控制无人机向右转
    参数:
        angle:旋转角度（度）
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode: 1/常亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
示例: api.single_fly_turnright(90)
     api.single_fly_turnright(90,{'r':16,'g':15,'b':100,'mode':1})
```

弹跳

```
single_fly_bounce(frequency, height,led)
'''
```

```
    描述:
        实时控制无人机弹跳
    参数:
        frequency:弹跳次数
        height:弹跳距离（厘米）
        led: 不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode: 1/常亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
'''
示例: api.single_fly_bounce(3, 50)
     api.single_fly_bounce(3, 50,{'r':16,'g':15,'b':100,'mode':1})
```

直线飞行

```
single_fly_straight_flight( x, y, z,speed, led)
'''
    描述:
        直线飞行(x,y,z)
    参数:
        x:坐标x（厘米）
        y:坐标y（厘米）
        z:坐标z（厘米）
        speed:不填默认为100 速度（0-100cm）/s
        led: 不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode: 1/常亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
示例: api.single_fly_straight_flight(100, 100, 100)
     api.single_fly_straight_flight(100, 100, 100,50,
{'r':16,'g':15,'b':100,'mode':1})
```

环绕飞行

```
single_fly_radius_around(radius,led)
'''
    描述:
        半径环绕飞行
    参数:
        radius: 环绕半径(厘米，正：逆时针 负：顺时针)
        led: 不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode: 1/常亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
示例: api.single_fly_radius_around(100)
     api.single_fly_radius_around(100,{'r':16,'g':15,'b':100,'mode':1})
```

自旋转

```
single_fly_autogyration360(num,led)
'''
    描述:
        顺时针、逆时针自转一定圈数
    参数:
        num:(正：逆时针 负：顺时针)
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode：1/常
亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
        '''
示例：api.single_fly_autogyration360(2)
     api.single_fly_autogyration360(2,{'r':16,'g':15,'b':100,'mode':1})
```

翻滚

```
single_fly_somersault(direction)
'''
    描述:
        无人机原地向前后左右翻滚
    参数:
        DIRECTION_FORWARD=0, /* forward. | */
        DIRECTION_BACK=1, /* back. | */
        DIRECTION_LEFT=2, /* left. | */
        DIRECTION_RIGHT=3, /* right. | */
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode：1/常
亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯

'''
示例：api.single_fly_somersault(0)
     api.single_fly_somersault(0,{'r':16,'g':15,'b':100,'mode':1})
```

曲线飞行

```
single_fly_curvilinearFlight(x, y, z, direction, speed, led)
'''
    描述:
        曲线飞行(x,y,z)
    参数:
        x：x轴坐标（厘米）（机体左右，右为正）
        y：y轴坐标（厘米）（机体前后，前为正）
        z：z轴坐标（厘米）（机体上下，上为正）
        direction：True：逆时针 False：顺时针 默认True
        speed:不填默认为100 速度（0-100cm）/s
        led：不填默认为0，格式:{'r':0,'g':0,'b':0,'mode':1}  r,g,b:色域，mode：1/常
亮，2/灭灯，4/RGB 三色循环，16/七彩灯,32/闪烁,64/呼吸灯
'''
示例：api.single_fly_curvilinearFlight(100, 100, 0, True, 50)
     api.single_fly_curvilinearFlight(100, 100, 0, False, 50,
{'r':16,'g':15,'b':100,'mode':1})
```

开启避障

```
single_fly_barrier_aircraft(mode)
'''
    描述:
    开启避障
    参数:
    mode:True:开启 False:关闭
'''
示例: api.single_fly_barrier_aircraft(True)
```

巡线检测

```
single_fly_Line_walking(fun_id, dist, way_color)
'''
    描述:
        巡线检测
    参数:
        fun_id  = 0  //0:向前巡线，无视路口
        dist          //距离，单位cm
        way_color   //巡线颜色色域，0-黑色 255-白色
        返回:
        return result = 1; //指令执行的结果：0-失败，1-成功 2-成功遇到路口

'''
示例: api.single_fly_Line_walking(0, 100, 0)
```

识别标签

```
single_fly_AiIdentifies(mode)
'''
    描述:
        识别标签
    参数:
        mode:0-9识别0-9的数字标签，10识别左箭头，11识别右箭头，12识别上箭头，13识别下箭头，20结束任务，65-90大写字母A-Z；触发识别后识别过程持续300ms，如果识别成功就立马结束
    返回:
        x;标签卡与无人机的X坐标
        y;标卡与无人机的Y坐标
        z;标卡与无人机的Z坐标
        angle;标卡与无人机的角度
        result; //False 识别失败，  True识别成功

'''
示例: api.single_fly_AiIdentifies(1)
```

光流对齐二维码

```
single_fly_Optical_flow_alignment(qr_id, qr_size, angle = 0)
'''
    描述:
        光流对齐二维码
    参数:
        qr_id; 到二维码id[0-9],
        qr_size:二维码的物理大小，范围[ 6, 30 ],默认值20，单位:cm
        angle:对齐二维码并旋转n度
    返回:
        result; //False 识别失败， True识别成功
'''
示例: api.single_fly_Optical_flow_alignment(1, 20, 0)
```

光流识别二维码

```
single_fly_Optical_flow_recognition(qr_id, qr_size)
'''
    描述:
        光流识别二维码
    参数:
        qr_id; 到二维码id[0-9],
        qr_size:二维码的物理大小，范围[ 6, 30 ],默认值20，单位:cm
    返回:
        {
            result; //False 识别失败， True识别成功
            x;//无人机与二维码之间的距离
            y;//无人机与二维码之间的距离
            z;//无人机与二维码之间的距离
            yaw;//无人机与二维码之间的角度
            qr_id;//识别到二维码的id
        }
'''
示例: api.single_fly_recognition_Qrcode(0, 1)
```

前摄对齐二维码

```
single_fly_Proactive_alignment(qr_id)
'''
    描述:
        前摄对齐二维码
    参数:
        qr_id; 到二维码id[0-9],
    返回:
        result; //False 识别失败， True识别成功
```

```
'''
示例: api.single_fly_Proactive_alignment(1)
```

前摄识别二维码

```
single_fly_Anticipatory_recognition(qr_id)
'''
    描述:
        前摄识别二维码
    参数:
        qr_id; 到二维码id[0-9]
    返回:
        {
            result; //False 识别失败， True识别成功
            x;//无人机与二维码之间的距离
            y;//无人机与二维码之间的距离
            z;//无人机与二维码之间的距离
            yaw;//无人机与二维码之间的角度
            qr_id;//识别到二维码的id
        }
'''
示例: api.single_fly_Anticipatory_recognition(1)
```

追踪二维码

```
single_fly_track_Qrcode(qr_id, time)
'''
    描述:
        追踪[0-9]号二维码[time]秒
    参数:
        qr_id:二维码id
        time:追踪时间
    返回:
        result: 0:成功，1:失败

'''
示例: api.single_fly_track_Qrcode(1, 10)
```

颜色识别，获取当前视频流一帧的颜色

```
single_fly_getColor()
'''
    描述:
        颜色识别，获取当前视频流一帧的颜色
    参数:
        Mode:1开始,跑一帧
```

```
    返回:
        r,g,b:色域
        state:0失败 1成功

'''
示例: ret = api.single_fly_getColor()#返回: r,g,b:色域
state:0失败 1成功
```

设置灯光颜色和模式,不会阻塞主线程

```
single_fly_lamplight(r, g, b, time, mode)
'''
    描述:
        设置灯光颜色和模式

    参数:
        r,g,b:色域
    time: 灯光时长/s
    mode: 1/常亮, 2/灭灯, 4/RGB 三色循环, 16/七彩灯,32/闪烁,64/呼吸灯
    返回:
        True:执行成功
        False:执行失败

'''
示例: api.single_fly_lamplight(255, 0, 0, 1, 1)#设置灯光颜色和模式
```

发射激光

```
plane_fly_generating(type, data ,reserve)
'''
    描述:
        发射激光
    参数:
        type = 0;  // 激光: 0-单发,1-连发, 2-开启激光接收, 3-关闭激光接收 4-一直连发无
弹量 5-关闭发射
        data = 10; // 激光连发频率, 次/秒, 范围1-14
        reserve = 100 //弹量,数据范围1-255
'''
示例: api.plane_fly_generating(0, 10, 100)#单发
     api.plane_fly_generating(2, 10, 100)开启激光接收
```

激光接收器被击中

```
plane_fly_laser_receiving()
'''
    描述:
```

```
        激光接收器被击中
    返回:
        True:被击中
        False:未击中
'''
示例: api.plane_fly_laser_receiving()
```

定位二维码开关

```
Plane_cmd_switch_QR(type)
'''
    描述:
        定位二维码开关
    参数:
        type:0-定位二维码开启 1-定位二维码关闭
'''
示例: api.Plane_cmd_switch_QR(0)
```

拍照

```
Plane_fly_take_photo()
'''
    描述:
    拍照,必须开启视频流后调用

'''
示例: api.Plane_fly_take_photo()#拍照
```

录像

```
Plane_cmd_switch_video(type)
'''
    描述:
        开始录像
    参数:
        type:// 录像，0-开启，1-结束

'''
示例: api.Plane_cmd_switch_video(0)#开启录像
```

开启视频流

```
Plane_cmd_swith_rtp(type)
'''
```

```
    描述:
    开启视频流
    参数:
    type:0-开启，1-关闭
'''
示例: api.Plane_cmd_swith_rtp(0)#开启视频流
```

打开视频流窗口

```
single_fly_flip_rtp()
'''
    描述:
    打开视频流(调用前需开启视频流)
    参数:

'''
示例: api.single_fly_flip_rtp()#打开视频流窗口
```

设置主摄俯仰角度

```
Plane_cmd_camera_angle(type, data)
'''
    描述:
    设置主摄俯仰角度
    参数:
    type = 0;  // 转动的方向: 0-上,1-下(绝对),2和3算法控制，4-校准，5-积木上，6-积木下
（相对）
    data = 30; // 转动的角度: 0~90

'''
示例: api.Plane_cmd_camera_angle(0, 30)#设置主摄俯仰角度
```

低速转动螺旋桨

```
plane_fly_arm()
'''
    描述:
    解锁电机
    参数:

'''
示例: api.plane_fly_arm()#低速转动螺旋桨
```

停止低速转动螺旋桨

```
plane_fly_disarm()
'''
    描述:
    关闭电机
    参数:

'''
示例: api.plane_fly_disarm()#停止低速转动螺旋桨
```

获取避障信息

```
Plane_getBarrier()
'''
    描述:
    获取避障信息
    参数:

    返回: 字典 每个方向的障碍物状态，True:有障碍物，False:无障碍物
    {
        'forward': True
        'back': True,
        'left': True,
        'right': True,
    }
'''
示例: ret = api.Plane_getBarrier()#获取避障信息
```

获取无人机电量百分比

```
get_battery()
'''
    描述:
    获取无人机电量百分比
    返回值:
    整数:电量百分比
'''
示例: ret = api.get_battery()#获取无人机电量百分比
```

获取无人机坐标(x,y,z)

```
get_coordinate()
'''
    描述:
    获取无人机坐标[x，y，z]
    参数:
```

```
    返回值:
    [x, y, z]
'''
示例: ret = api.get_coordinate()#获取无人机坐标[x, y, z]
```

获取无人机角度

```
get_yaw()
'''
    描述:
        获取无人机角（度）
    返回值:
        整数:[偏航角,俯仰角，翻滚角]
'''
示例: ret = api.get_yaw()
```

获取无人机机体速度(X轴速度,Y轴速度,Z轴速度)

```
get_plane_speed()
'''
    描述:
        获取无人机机体速度(X轴速度,Y轴速度,Z轴速度)
    返回值:
        整数:[X,Y,Z]
'''
示例: ret = api.get_plane_speed()
```

获取无人机Tof高度

```
get_plane_distance()
'''
    描述:
       获取无人机Tof高度
    返回值:
       整数:无人机Tof高度
'''
示例: ret = api.get_plane_distance()
```

获取无人机Tof高度

```
get_plane_id()
'''
    描述:
```

```
        获取无人机ID
    返回值:
        整数:无人机ID
'''
示例: ret = api.get_plane_id()
```

外挂电磁铁

```
Plane_cmd_electromagnet(type)
'''
    描述:
        外挂电磁铁
    参数:
        type:2-电磁铁吸附 3-电磁铁弹出
'''
示例: ret = api.Plane_cmd_electromagnet(2)
```

外挂夹子,电磁铁

```
Plane_cmd_clamp(type,angle = 0)
'''
    描述:
        外挂夹子,电磁铁
    参数:
        type:0:夹子失能,1:夹子使能,2:夹子角度,3:电磁铁弹出,4:电磁铁吸附
        angle:夹子转动到角度 0-180

'''
示例:    api.Plane_cmd_clamp(0)##夹子失能
        api.Plane_cmd_clamp(1)##夹子使能
        api.Plane_cmd_clamp(2,30)##要先调用夹子使能 夹子角度开启30度
        api.Plane_cmd_clamp(3)##电磁铁弹出
        api.Plane_cmd_clamp(4)##电磁铁吸附
```