# サンプルプログラムの解説

```
import cv2
from dynamixel_sdk import *
from dynamixel_sdk import Protocol2PacketHandler as P2PH                    # Uses
Dynamixel SDK library
from dynamixel_sdk import PortHandler
from dynamixel_sdk import PacketHandler
```

赤字部分が dynamixel_sdk のインポートである。

ここでエラーが出ている場合、パスが通っていないためフォルダの場所を確認する

```
# Control table address
AX_TORQUE_ENABLE          = 24 # Control table address is different in Dynamixel model
AX_GOAL_POSITION          = 30
AX_PRESENT_POSITION       = 36
AX_MOVING_SPEED           = 32
AX_MOVING                 = 46
# Protocol version
PROTOCOL_VERSION          = 1 # See which protocol version is used in the Dynamixel

# Default setting
DXL_ID          = 1                                 # Dynamixel ID: 1
BAUDRATE        = 1000000
DEVICENAME          = "COM3"

TORQUE_ENABLE           = 1
TORQUE_DISABLE          = 0
DXL_MINIMUM_POSITION_VALUE=0
DXL_MAXIMUM_POSITION_VALUE    = 1023
ESC_ASCII_VALUE                 = 0x1b

COMM_SUCCESS                    = 0                                 # Communication
Success result value
COMM_TX_FAIL                    = -1001                             # Communication
Tx Failed
```

ロボットで実装したプログラムの#define と同様初期定義である。

ここで注意する点は赤字で記述してある DEVICENAME である。

これは、Dynamixel を pc につないだポート番号に設定する必要がある。

よって各自で COM 番号を確認し変更する。

```
flag = 0
```

flag はグローバル変数である。参照するさいに注意する必要がある。
関数などの参照できないスコープからグローバル変数を呼び出す場合は、
global flag と記述し、グローバル変数の flag であることを明示しなければならない。

```
cap = cv2.VideoCapture(0) #camera の設定
```

カメラの設定である。ノート pc に付属のカメラの場合は引数に 0 を指定する。

```
class DXL():
    def __init__(self):
        # Open port
        if portHandler.openPort():
            print("Succeeded to open the port")
        else:
            print("Failed to open the port")
            print("Press any key to terminate...")
            quit()
        # Set port baudrate
        if portHandler.setBaudRate(BAUDRATE):
            print("Succeeded to change the baudrate")
        else:
            print("Failed to change the baudrate")
            print("Press any key to terminate...")
            quit()
        # Enable Dynamixel Torque
        dxl_comm_result, dxl_error = packetHandler.write1ByteTxRx(portHandler,
DXL_ID, AX_TORQUE_ENABLE, TORQUE_ENABLE)
        if dxl_comm_result != COMM_SUCCESS:
            print("%s" % packetHandler.getTxRxResult(dxl_comm_result))
        elif dxl_error != 0:
            print("%s" % packetHandler.getRxPacketError(dxl_error))
        else:
            print("Dynamixel has been successfully connected")
```

クラスの定義。

Def init()メソッドではコンストラクタの役割を行う。プログラム実行時、一度だけ実行される。一度だけ設定を行う必要のあるプログラムはここに追加するとよい。また、dynamixel のトルクの on/off 設定も行っているため、各自用いる dynamixel のトルク設定は追加する必要がある。

```
  def moveDXL(self):
            global flag
            # Read moving state

dxl_moving_state,dxl_comm_result,dxl_error=packetHandler.read1ByteTxRx(portHandler,
DXL_ID, AX_MOVING)
            # Write goal position
            if dxl_moving_state == 0:
                if flag == 1:
                    dxl_comm_result, dxl_error = packetHandler.write2ByteTxRx(portHandler,
DXL_ID, AX_GOAL_POSITION, DXL_MINIMUM_POSITION_VALUE)
                    flag=0
                else:
                    dxl_comm_result, dxl_error = packetHandler.write2ByteTxRx(portHandler,
DXL_ID, AX_GOAL_POSITION, DXL_MAXIMUM_POSITION_VALUE)
                    flag = 1
```

def moveDXL()では関節モードの dynamixel を角度 0～300° まで連続して動作させている。
ここで、グローバル変数の flag を用いている理由は、角度の最小値 0 と最大値 1023 を交互に実行するため、if 分にて flag を用いて場合分けを行っている。
また、最も重要なのが AX_MOVING にて dynamixel の動作命令が実行完了しているかを確認している点である。動作命令の実行が完了された場合のみ次の動作命令を実行するように dxl_moving_state にて場合分けしている。

車輪モードの場合は AX_MOVING_SPEED をアドレス番号の引数として指定することで、動作命令を送れる。

```
if __name__ == "__main__":
```

```
    dx = DXL()

    try:
        while True:
            ret, frame = cap.read()#　カメラの読み込み
            img = frame
            print("press ESC to quit!")
            cv2.namedWindow('Camera', cv2.WINDOW_AUTOSIZE)
            cv2.imshow('Camera',img)
            if cv2.waitKey(1) & 0xff == 27:
                break

            dx.moveDXL()# dynamixel を動かすメソッド

##########################################################################

    finally:
        # Close port
        cap.release()
        cv2.destroyAllWindows()
        portHandler.closePort()
```

このプログラムのメインループである。dx = DXL()にてクラスのインスタンス化を行い、dx.でクラスのメソッドを参照可能となる。
ret, frame = cap.read()にてカメラ画像を読み込み、変数に格納している。
画像処理を行う場合は、この frame をあつかうと良い。Esc キーを入力すると finally の close メソッドたちが実行され、プログラムが終了する。

dx.moveDXL()このように、クラス DXL()の moveDXL()メソッドを呼び出すことができる。