메인프로그램 SubMarine.py

SubMarine.py

메인 프로그램

```
from flask import Flask, render_template, send_from_directory, Response, send_file,
request, redirect, url_for
from flask_socketio import SocketIO
import argparse, logging, logging.config, conf
import os
from power import PowerStatus
from datetime import datetime
from commChecker import CommChecker
from gpioControl import GpioControl
from usbwebcamera import UsbWebCamera
from piwebcamera import PiWebCamera


app = Flask(__name__)
app.config['SECRET_KEY'] = '1764121q'
socketio = SocketIO(app)

logging.config.dictConfig(conf.dictConfig)
logger = logging.getLogger(__name__)

power = PowerStatus()
commChecker = CommChecker()
gpio = GpioControl()

usbcamera = UsbWebCamera(20,1)
usbcamera.run()

piwebcamera = PiWebCamera()
piwebcamera.run()

@app.after_request
def add_header(r):
    """
    Add headers to both force latest IE rendering or Chrome Frame,
    and also to cache the rendered page for 10 minutes
    """
    r.headers["Cache-Control"] = "no-cache, no-store, must-revalidate"
    r.headers["Pragma"] = "no-cache"
    r.headers["Expires"] = "0"
    r.headers["Cache-Control"] = "public, max-age=0"
```

```
    return r

@app.route("/")
@app.route("/index.html")
def index():
    logger.debug("Requested /")
    return render_template("index.html")

@app.route("/favorit.ico")
def favorit_ico():
    logger.debug("Requested favorit.ico image")
    filename = "favorit.ico"
    return send_file(filename)

@app.route("/VideoStatus")
def videostatus():
    logger.debug("Video Status change ")
    content = usbcamera.status()
    return Response(content, mimetype='text/xml')

@app.route("/SubmStatus")
def submStatus():
    logger.debug("Submarine Status change ")
    content = piwebcamera.status()
    return Response(content, mimetype='text/xml')

@app.route("/temperature")
def temperature():
    content = os.popen("vcgencmd measure_temp").readline()
    content = content.replace("temp=", "")
    powerstatus = power.getPowerStatus()
    return Response("내부온도:" + content+"  , 전원:"+powerstatus+" , 측정시
간:"+str(datetime.now()), mimetype='text/xml')

@app.route("/commCheck")
def commcheck():
    connstatus = str(commChecker.checkQuality())
    return Response(connstatus+" , "+str(datetime.now()), mimetype='text/xml')

@app.route('/lightOnOff', methods=['GET', 'POST'])
def lightOnOff():
    gpio.light()
    logger.debug('message KK Fired !!!')
    return Response('Light ON', mimetype='text/html')

''' ##############   MOVE    #############  '''

@app.route("/floating", methods=['GET', 'POST'])
def floating():
    if request.method == 'POST':
```

```
        moveOrder = request.form['moveCnt']
        if not moveOrder :
            reString = "Do not move"
        else :
            reString = gpio.Floating(int(moveOrder))
        print("Move floating ..." + moveOrder)
    return Response(reString, mimetype='text/html')

@app.route("/diving", methods=['GET', 'POST'])
def diving():
    if request.method == 'POST':
        moveOrder = request.form['moveCnt']
        if not moveOrder :
            reString = "Do not move"
        else :
            reString = gpio.Diving(int(moveOrder))
        print("Move diving ..." + moveOrder)
    return Response(reString, mimetype='text/html')



@app.route("/forward", methods=['GET', 'POST'])
def forward():
    if request.method == 'POST':
        moveOrder = request.form['moveFBLFCnt']
        if not moveOrder :
            reString = "Do not move"
        else :
            reString = gpio.Forward(int(moveOrder))
        print("Move Forward ..." + moveOrder)
    return Response(reString, mimetype='text/html')

@app.route("/backward", methods=['GET', 'POST'])
def backward():
    if request.method == 'POST':
        moveOrder = request.form['moveFBLFCnt']
        if not moveOrder :
            reString = "Do not move"
        else :
            reString = gpio.Backward(int(moveOrder))
        print("Move Backward ..." + moveOrder)
    return Response(reString, mimetype='text/html')

@app.route("/right", methods=['GET', 'POST'])
def right():
    if request.method == 'POST':
        moveOrder = request.form['moveFBLFCnt']
        if not moveOrder :
            reString = "Do not move"
        else :
```

```
            reString = gpio.TurnLeftF(int(moveOrder))
        print("Move Right ..." + moveOrder)
    return Response(reString, mimetype='text/html')

@app.route("/left", methods=['GET', 'POST'])
def left():
    if request.method == 'POST':
        moveOrder = request.form['moveFBLFCnt']
        if not moveOrder :
            reString = "Do not move"
        else :
            reString = gpio.TurnRightF(int(moveOrder))
        print("Move Left ..." + moveOrder)
    return Response(reString, mimetype='text/html')


@app.route("/video_usb_feed")
def video_usb_feed():
    return Response(genusb(usbcamera),
        mimetype="multipart/x-mixed-replace; boundary=frame")

def genusb(usbcamera):
    logger.debug("Starting USB stream")
    while True:
        frame = usbcamera.get_frame()
        yield (b'--frame\r\n'
               b'Content-Type: image/png\r\n\r\n' + frame + b'\r\n')


@app.route("/video_pi_snapshot")
def video_pi_snapshot():
    return Response(genpi(piwebcamera),
        mimetype="multipart/x-mixed-replace; boundary=frame")

def genpi(piwebcamera):
    logger.debug("Starting PI capchure")
    while True:
        frame = piwebcamera.get_frame()
        yield (b'--frame\r\n'
               b'Content-Type: image/png\r\n\r\n' + frame + b'\r\n')


if __name__=="__main__":
    logger.debug("Starting Submarine start")
#    socketio.run(app, log_output=True, host='0.0.0.0', port=8081, debug=True,
use_reloader=False, allow_unsafe_werkzeug=True)
    socketio.run(app, log_output=True, host='0.0.0.0', port=8081,
allow_unsafe_werkzeug=True)
```

Revision #2
Created 2024-10-01 13:54:54 UTC by Hyeon Su Ryu
Updated 2024-10-01 13:57:50 UTC by Hyeon Su Ryu