Appendix1 ↓

```
from m5stack import lcd
import utime
import wifiCfg
import hat
import urequests
import ujson
import gc

# チャネル(bucket)
CHANNEL = "CQCH1"
# measurement
MEASUREMENT = "M5StickC"

#データサーバのアドレス
DATA_SERVER = "http://192.168.3.3:8000/data/" + CHANNEL + "/" + MEASUREMENT

#データ送信間隔(秒)
INTERVAL = 60

#画面クリア
lcd.clear()
lcd.setRotation(1)
lcd.setCursor(0, 0)

# WiFi接続
wifiCfg.autoConnect()

header = {'Content-Type' : 'application/json'}

# HAT ENV IIを使って環境データを取得してデータ・サーバに送信する。
hat_env2 = hat.get(hat.ENV2)
while True:
        # 気温、湿度、気圧を取得
        temp = hat_env2.temperature
        hum = hat_env2.humidity
        pre = hat_env2.pressure

        # 表示
        lcd.clear()
        lcd.setCursor(0, 0)
        wait_ms(2)
        lcd.println('Temp:' + str(("%.2f"%temp)) + ' deg.')
```

```
    lcd.println('Humidity:' + str(("%.2f"%hum)) + '%')
    lcd.println('Pres:' + str(("%.2f"%pre)) + ' hPa')
    wait(1)

    # データ・サーバにデータを送信
    dp = {
        "source":"M5StickC Plus",
        "data":[{"temperature":temp},{"pressure":pre},{"humidity":hum}]
        }

    #resp = urequests.post(DATA_SERVER, data = {"source":"A"}, headers = header)
    resp = urequests.post(DATA_SERVER, data = ujson.dumps(dp).encode("utf-8"), headers = header)
    lcd.println(str(resp.status_code))

    gc.collect()

    wait(INTERVAL)

```

Appendix1　リスト1

Appendix2↓

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-


import time
import sys
import json
import libdata
import traceback
import BME280.bme280_sample as MySensor

# configファイルのロード
def load_config(path='./config.json'):
    global config
    f = open(path, 'r')
    config = json.load(f)
    f.close()
    print(config)

```

```
load_config('./config.json')
libdata.set_url("http://" + config['server'] + ":" + str(config['port']))

# チャネル(bucket)
CHANNEL = config['channel']

# measurement
MEASUREMENT = config['measurement']

# データ送信間隔
INTERVAL = 60

# データ取得・送信を無限ループ
while True:
    try:
        # データを読み込む

        data = MySensor.get_data()
        print(data)

        # データを保存

        libdata.send_data(CHANNEL, MEASUREMENT, "raspi", data)

    except KeyboardInterrupt:
        break
    except:
        print(traceback.format_exc())
        pass


    # 指定時間待ち

    time.sleep(INTERVAL)


```

Appendix2　リスト1

```
#!/usr/bin/env python
# -*- coding: utf-8 -*-

import time
```

```
import sys
import requests
import json

config = {}

# 初期化
api_url = "http://192.168.3.3:8000"
def set_url(url):
    global api_url
    api_url = url
    return



# データ保存
def send_data(channel, measurement, source, data):
    try:
        url = api_url + "/data/" + channel + "/" + measurement
        data = {"source":source, "data":data}
        resp = requests.post(url, json=data)
        r = resp.status_code
    except Exception as e:
        r = 500
        print(sys._getframe().f_code.co_name+":post failed")
        print(e)

    return r


# ファイル保存
def send_file(channel, name, path):
    try:
        url = api_url + "/data/" + channel
        resp = requests.post(url, files={'file':(path, open(path, 'rb'))}, data={'name': name})
        r = resp.status_code

    except Exception as e:
        r = 500
        print(sys._getframe().f_code.co_name+":post failed")
        print(e)

    return r

```

```
 49
 50
 51  if __name__ == '__main__':
 52
 53      load_config('./config.json')
 54      set_url("http://" + config['server'] + ":" + str(config['port']))
 55      data = [{"temperature":38.0},{"pressure":1000},{"humidity":90}]
 56      send_data("mybucket","dev1m", "raspi", data)
 57      send_file("mybucket", "myfile", "/tmp/test.file")
 58
 59
```

Appendix2　リスト2

```
  1  #coding: utf-8
  2
  3  from smbus import SMBus
  4  import time
  5  import json
  6  import traceback
  7  import time
  8
  9  bus_number  = 1

  ～～　省略　～～
 64  def readData():

 ～～　省略　～～

 71
 72          t = compensate_T(temp_raw)
 73          p = compensate_P(pres_raw)
 74          h = compensate_H(hum_raw)
 75          return t,p,h

 ～～　省略　～～

 77  def compensate_P(adc_P):

 ～～　省略　～～

 99          print "pressure : %7.2f hPa" % (pressure/100)
100          return pressure/100
```

```
   ～～  省略  ～～

   102  def compensate_T(adc_T):

   ～～  省略  ～～

   108          print "temp : %-6.2f ℃" % (temperature)
   109          return temperature

   ～～  省略  ～～

   111  def compensate_H(adc_H):

   ～～  省略  ～～

   123          print "hum : %6.2f %" % (var_h)
   124          return var_h

   ～～  省略  ～～

   144  def makeres(t,p,h):
   145          dp = [{"temperature":t}, {"pressure":p}, {"humidity":h}]
   146          return dp
   147

   ～～  省略  ～～

   153  def get_data():
   154          t,p,h = readData()
   155          return makeres(t,p,h)
   156
```

Appendix2　リスト3