M5Paper 環境モニターソース

美都

2021 年 3 月 17 日

目次

1 メイン

```
#include <M5EPD.h>
#define LGFX_M5PAPER
#include <LovyanGFX.hpp>

#include "battery.h"
#include "thermometer.hpp"
#include "infoFromNet.hpp"
#include "tokei.hpp"

static LGFX lcd;

void drawLcd() {
    drawBattery(960-120-5, 5, &lcd);
    Thermometer t = Thermometer(200,200);
    t.drawTempMeter(&lcd, 500, 100);
    t.drawHumMeter(&lcd, 700, 100);

    Tokei tokei = Tokei(300, 100);
    tokei.drawDigitalTokei(&lcd, 100, 100);

    delay(500);
}

// ●分ピッタリまでの秒数
int rest_minute() {
    rtc_time_t time;
    M5.RTC.getTime(&time);
    return 60-time.sec;
}

// シャットダウンを試みる。通電中はすり抜ける
void challengeShutdown() {
    int rest_sec = rest_minute()-6;
    if (rest_sec < 30) rest_sec += 60;
    M5.shutdown(rest_sec); // 一旦停止
}

void checkInfoFromNetwork(bool always=false) {
    rtc_time_t time;
    rtc_date_t date;
    M5.RTC.getDate(&date);
    M5.RTC.getTime(&time);
    if ((time.hour%6==0 && time.min<1) || date.year<2020 || always) {
        Serial.println("ネットワークの情報の取得開始");
        GetInfoFromNetwork info;
        info.setNtpTime();
        info.getWeatherInfo();
    }
}

void setup()
{
    M5.begin(false, true, true, true, true);
    M5.BatteryADCBegin();
```

```
    M5.RTC.begin();
    M5.SHT30.Begin();
    SD.begin();
    Serial.begin(115200);
    lcd.init();
    lcd.setRotation(1);

    checkInfoFromNetwork();

    drawLcd();
    challengeShutdown();
}


void loop()
{
    delay((rest_minute()+1)*1000);
    checkInfoFromNetwork();
    drawLcd();
    challengeShutdown();
}
```

2 バッテリーメーター

2.1 battery.h

```
#include <M5EPD.h>
#define LGFX_M5PAPER
#include <LovyanGFX.hpp>

// バッテリー残量を(x,y)に表示する。
int drawBattery(int x, int y, LGFX *lcd) ;
int get_rest_battery() ;
```

2.2 battery.cpp

```
#include "battery.h"

// バッテリー残量の取得
int get_rest_battery() {
    const int max_vol = 4350;
    const int min_vol = 3300;
    //M5.BatteryADCBegin();
    int voltage = M5.getBatteryVoltage();
    voltage = max(voltage, min_vol);
    voltage = min(voltage, max_vol);
    float rest_battery_raw = (float)(voltage - min_vol) / (float)(max_vol - min_vol);
    rest_battery_raw = max(rest_battery_raw, 0.01f);
    rest_battery_raw = min(rest_battery_raw, 1.f);
    return (int)(rest_battery_raw * 100);
}

// バッテリー残量計の表示
int drawBattery(int x, int y, LGFX *lcd) {
    LGFX_Sprite battery_meter(lcd);
    int rest_battery = get_rest_battery();

    // バッテリー矩形の表示
    battery_meter.setColorDepth(4);
    battery_meter.createSprite(120, 30);
    battery_meter.fillSprite(15);
    battery_meter.setColor(0);
    battery_meter.drawRect(10, 10, 45, 20);
    battery_meter.fillRect(55, 17, 5, 5);
    battery_meter.fillRect(10, 10, (int)((45*rest_battery)/100), 20);

    // バッテリー残量文字の表示
    battery_meter.setFont(&fonts::lgfxJapanMinchoP_20);
    battery_meter.setTextSize(1, 1); // 縦,横　倍率
    battery_meter.setTextColor(0, 15); // 文字色,背景
    battery_meter.setCursor(62, 10);
    battery_meter.printf("%d%%", rest_battery);

    lcd->startWrite();
    battery_meter.pushSprite(x, y);
    lcd->endWrite();

```

```
42      return rest_battery;
43  }
```

3 温湿度計

3.1 thermometer.hpp

```
#include <M5EPD.h>
#define LGFX_M5PAPER
#include <LovyanGFX.hpp>

class Thermometer {
    private:
        float temp;
        float hum;
        int sizex;
        int sizey;
        float radius;
        LGFX_Sprite face;
        LGFX_Sprite scale[2];
        LGFX_Sprite hand;

        void makeMeterFace(int min, int max, const char* unit);
        void makeScale();
        void makeHand();
    public:
        Thermometer(int sizex=200, int sizey=200);

        float get_temp();
        float get_hum();

        void drawTempMeter(LovyanGFX *lcd, int x, int y );
        void drawHumMeter(LovyanGFX *lcd, int x, int y);
        void drawString(LovyanGFX *lcd, int x, int y);
};
```

3.2 thermometer.cpp

```
#include "thermometer.hpp"

Thermometer::Thermometer(int sizex, int sizey)
    : sizex(sizex), sizey(sizey) {
    M5.SHT30.Begin();
    radius = min(sizex, sizey)/2*0.95;
    makeScale();
    makeHand();
};

void Thermometer::makeScale() {
    scale[0].setColorDepth(4);
    scale[0].createSprite(radius/25, radius/5);
    scale[0].fillSprite(0);
    scale[0].setPivot(scale[0].width()/2, scale[0].height());
    scale[1].setColorDepth(4);
    scale[1].createSprite(radius/40, radius/7);
    scale[1].fillSprite(0);
    scale[1].setPivot(scale[1].width()/2, scale[1].height());
}
```

```
21
22  void Thermometer::makeHand() {
23      float height, width;
24      height = radius * 0.8f;
25      width = height * 0.1f;
26
27      hand.setColorDepth(4);
28      hand.createSprite(width, height);
29      hand.fillSprite(15);
30      hand.setColor(0);
31      hand.fillTriangle(width/2.f, 0, 0, height/4.f, width, height/4.f);
32      hand.fillTriangle(0, height/4.f, width, height/4.f, width/2.f, height);
33      hand.setPivot(width/2., height);
34  }
35
36  void Thermometer::makeMeterFace(int min, int max, const char* unit) {
37      face.setColorDepth(4);
38      face.createSprite(sizex, sizey);
39      face.fillSprite(15);
40      face.setColor(0);
41      face.setFont(&fonts::lgfxJapanGothic_36);
42      face.setTextColor(0, 15);
43      face.setTextDatum(middle_center);
44      float center[2] = {sizex/2.0f, sizey/2.0f};
45      face.fillCircle(center[0], center[1], radius);
46      face.fillCircle(center[0], center[1], radius*0.95, 15);
47      float angleInterval = 270.f / (float)(max-min);
48      for (int i = min ; i <= max ; i+=2) {
49          LGFX_Sprite *use_scale = (i%10==0) ? &scale[0] : &scale[1];
50          float angle = (270.f-45.f) - (float)(i-min) * angleInterval;
51          float angleRad = angle * 3.14159265f / 180.f ;
52          float startx = (radius - use_scale->height()) * cos(angleRad) + center[0];
53          float starty = -1.0f * ((radius - use_scale->height()) * sin(angleRad)) + center[1];
54          use_scale->pushRotateZoom(&face, startx, starty, 90.f-angle, 1.f, 1.f);
55          if (i%10==0) {
56              float charsize = (float)scale[0].height() / 36.f;
57              float charx = (radius - use_scale->height() * 1.5f) * cos(angleRad) + center[0];
58              float chary = -1.f * (radius - use_scale->height() * 1.5f) * sin(angleRad) +
                  center[0];
59              face.setTextSize(charsize);
60              face.drawNumber(i, charx, chary);
61          }
62          face.drawString(unit, center[0], sizey/5.f*3.f);
63      }
64
65  }
66
67  void Thermometer::drawTempMeter(LovyanGFX *lcd, int x, int y) {
68      makeMeterFace(0, 50, "℃");
69      float center[2] = {(float)sizex/2.f, (float)sizey/2.f};
70      float angle = 270.f - 45.f;
71      angle -= 270.f / 50.f * get_temp();
72      hand.pushRotateZoom(&face, center[0], center[1], 90.f - angle, 1.f, 1.f);
73      face.pushSprite(lcd, x, y);
74  }
75
76  void Thermometer::drawHumMeter(LovyanGFX *lcd, int x, int y) {
77      makeMeterFace(20, 80, "%");
```

```
    float center[2] = {(float)sizex/2.f, (float)sizey/2.f};
    float angle = 270.f - 45.f;
    angle -= 270.f / 60.f * (get_hum() - 20.f);
    hand.pushRotateZoom(&face, center[0], center[1], 90.f - angle, 1.f, 1.f);
    face.pushSprite(lcd, x, y);
}

void Thermometer::drawString(LovyanGFX *lcd, int x, int y) {
    M5.SHT30.UpdateData();
    LGFX_Sprite meter(lcd);
    meter.setColorDepth(4);
    meter.createSprite(250, 100);
    meter.fillSprite(15);
    meter.setColor(0);
    meter.setTextColor(0, 15);
    meter.setFont(&fonts::lgfxJapanMinchoP_36);
    meter.setCursor(10,10);
    meter.printf("温度:%5.1f℃", this->get_temp());
    meter.setCursor(10,50);
    meter.printf("湿度:%5.1f%%", this->get_hum());
    meter.pushSprite(x, y);
}

float Thermometer::get_temp() {
    M5.SHT30.UpdateData();
    this->temp = M5.SHT30.GetTemperature();
    return this->temp;
}

float Thermometer::get_hum() {
    M5.SHT30.UpdateData();
    this->hum = M5.SHT30.GetRelHumidity();
    return this->hum;
}
```

4 ネットからの情報取得

4.1 infoFromNet.hpp

```
// ネットワークより取得する情報関連
// 時計・天気予報
#include <M5EPD.h>

#define WeatherFileName "/weather.jsn"

class GetInfoFromNetwork {
    private:

        bool wifiOn(void);
        void wifiOff(void);
    public:
        GetInfoFromNetwork();
        ~GetInfoFromNetwork() ;
        int isWiFiOn(void);
        int setNtpTime() ;
        String getWeatherQuery() ;
        bool getWeatherInfo() ;
};
```

4.2 wifiid.h

```
// wifiのssdiとパスワード
#define ssid "LAKINET"
#define password "mitomiilaki31412101218"
```

4.3 apikey.h

```
#define apikey "b957044d4e4e6a5b1bfb05fc0ecf9255"
```

4.4 infoiFromNet.cpp

```
#include <M5EPD.h>
#include <WiFi.h>
#include <HTTPClient.h>
#include "infoFromNet.hpp"
#include <time.h>


// wifiid.hには、ssid,passwordの各defineを定義を文字列として記載すること。
// このファイルは、.gitignoreとする。
#include "wifiid.h"
// apikey.hには、apikeyのdefineの定義を文字列として記載すること。
// このファイルは、.gitignoreとする。
#include "apikey.h"

GetInfoFromNetwork::GetInfoFromNetwork() {
```

```
16      wifiOn();
17  }
18
19  GetInfoFromNetwork::~GetInfoFromNetwork() {
20      wifiOff();
21  }
22
23  bool GetInfoFromNetwork::wifiOn(void) {
24      WiFi.begin(ssid, password);
25      for (int i = 0 ; i < 10; i++) {
26          if (isWiFiOn()) return true;
27          delay(500);
28      }
29      return false;
30  }
31
32  void GetInfoFromNetwork::wifiOff(void) {
33      WiFi.disconnect(true);
34      WiFi.mode(WIFI_OFF);
35  }
36
37  int GetInfoFromNetwork::isWiFiOn(void) {
38      return (WiFi.status() == WL_CONNECTED) ;
39  }
40
41  int GetInfoFromNetwork::setNtpTime() {
42      if (!isWiFiOn()) return -1;
43      const long gmtOffset_sec = 9 * 3600;
44      const int daylightOffset_sec = 0;
45      const char * ntpServer = "jp.pool.ntp.org";
46
47      configTime(gmtOffset_sec, daylightOffset_sec, ntpServer);
48      struct tm timeinfo;
49      if (!getLocalTime(&timeinfo)) return -1;
50
51      rtc_time_t rtcTime;
52      rtcTime.hour = (int8_t)timeinfo.tm_hour;
53      rtcTime.min = (int8_t)timeinfo.tm_min;
54      rtcTime.sec = (int8_t)timeinfo.tm_sec ;
55      rtc_date_t rtcDate ;
56      rtcDate.year = (int8_t)timeinfo.tm_year + 1900;
57      rtcDate.mon = (int8_t)timeinfo.tm_mon + 1;
58      rtcDate.day = (int8_t)timeinfo.tm_mday ;
59      M5.RTC.setDate(&rtcDate);
60      M5.RTC.setTime(&rtcTime);
61      return 0;
62  }
63
64
65  const uint8_t fingerprint[20] =
66      { 0xEE,0xAA,0x58,0x6D,0x4F,0x1F,0x42,0xF4,0x18,0x5B,0x7F,0xB0,0xF2,0x0A,0x4C,0xDD,0x97,0
        x47,0x7D,0x99 };
67  #define OpenWeatherUrl "api.openweathermap.org"
68  #define City "Nagahama,JP"
69
70  String GetInfoFromNetwork::getWeatherQuery() {
71      String url("/data/2.5/forecast?");
72      url += "q=" City;
```

```
73      url += "&appid=" apikey;
74      url += "&lang=ja&units=metric";
75      return url;
76  }
77
78  // 天気予報データをSDカードのWeatherFileNameに書き込む。
79  bool GetInfoFromNetwork::getWeatherInfo() {
80      if (!isWiFiOn()) return false;
81      HTTPClient http;
82      File file;
83      String url = String("http://") + String(OpenWeatherUrl) + getWeatherQuery();
84      Serial.print("URL:");
85      Serial.println(url);
86      if (!http.begin(url)) return false;
87      int retCode = http.GET();
88      if (retCode < 0) goto http_err;
89      if (retCode != HTTP_CODE_OK && retCode != HTTP_CODE_MOVED_PERMANENTLY) goto http_err;
90      if (!SD.exists("/")) goto http_err;
91      Serial.println("SD OK!");
92      if (SD.exists(WeatherFileName)) SD.remove(WeatherFileName);
93      file=SD.open(WeatherFileName, FILE_WRITE);
94      if (!file) goto http_err;
95      Serial.println("ファイルオープン完了");
96      if (http.writeToStream(&file) < 0) goto file_err;
97      file.close();
98      Serial.println("weatherファイルへのjsonデータ書き込み完了");
99      http.end();
100     return true;
101
102 file_err:
103     file.close();
104 http_err:
105     http.end();
106     return false;
107 }
```

5 時計の表示

5.1 tokei.hpp

```
/************************************************
 * 時計の表示
 ************************************************/

#include <M5EPD.h>
#define LGFX_M5PAPER
#include <LovyanGFX.hpp>

class Tokei {
    private:
        int year, month, day;
        int hour, min, sec;
        int dayOfTheWeek;
        int width, height;

        void getDateTime();
        int getDayOfTheWeek(int year, int month, int day) ;

    public:
        Tokei(int sizex=200, int sizey=200);
        void drawDigitalTokei(LovyanGFX *lcd, int x, int y);
};
```

5.2 tokei.cpp

```
/************************************************
 * 時計の表示
 ************************************************/

#include <M5EPD.h>
#define LGFX_M5PAPER
#include <LovyanGFX.hpp>

#include "tokei.hpp"

Tokei::Tokei(int width, int height)
        : width(width), height(height) {
    getDateTime();
    dayOfTheWeek = getDayOfTheWeek(year, month, day);
}

// RTCより現在時刻を取得する。
void Tokei::getDateTime() {
    rtc_time_t time;
    rtc_date_t date;

    M5.RTC.getTime(&time);
    M5.RTC.getDate(&date);

    year = date.year;
    month = date.mon;
```

```
27      day = date.day;
28      hour = time.hour;
29      min = time.min;
30      sec = time.sec;
31  }
32
33  // 曜日の計算。月曜日を0、日曜日を6とする。
34  int Tokei::getDayOfTheWeek(int year, int month, int day) {
35      int y = year % 100;
36      int c = y / 100;
37      int ganma = 5 * c + c / 4;
38      return (day+(26*(month+1))/10+y+y/4+ganma+5)%7;
39  }
40
41  // デジタル時計を描画する
42  void Tokei::drawDigitalTokei(LovyanGFX *lcd, int x, int y) {
43      // 描画領域区分比率
44      const float tokei_ratio = 0.75f;
45      // スプライト初期化
46      LGFX_Sprite tokei;
47      tokei.setColorDepth(4);
48      tokei.createSprite(width,height);
49      tokei.fillSprite(15);
50      //tokei.drawRect(0,0,width,height,0); // レイアウト検討用外枠
51      // 時計時間部分表示
52      char strTime[6];
53      sprintf(strTime, "%02d:%02d", hour, min);
54      tokei.setFont(&fonts::Font7); // font高さ:48
55      tokei.setTextColor(0,15);
56      float mag = (height*tokei_ratio) / 48.f;
57      tokei.setTextSize(mag, mag);
58      tokei.drawString(strTime, 0.f, height*(1-tokei_ratio));
59      // 時計日時部分表示
60      const char* youbi_tbl[] =
61          { "月", "火", "水", "木", "金", "土", "日" };
62      const char* youbi = youbi_tbl[dayOfTheWeek];
63      char strDate[30];
64      sprintf(strDate, "%d年%2d月%2d日(%s)", year, month, day, youbi);
65      tokei.setFont(&fonts::lgfxJapanGothic_36);
66      mag = (height*(1-tokei_ratio)) / 36.f;
67      tokei.setTextSize(mag, mag);
68      tokei.drawString(strDate, 0.f, 0.f);
69      tokei.pushSprite(lcd, x, y);
70  }
```