```
/*
ESP32にI2C接続した温度計をブラウザに表示する。自動リフレッシュで3秒おきに表示する。
温度センサー：STS21(センシリオン)
*/


// Wi-Fiを使うためWiFi.hをインクルード
#include <WiFi.h>

// I2Cを使うためWire.hをインクルード
#include <Wire.h>

// Wi-Fi接続情報を入力(IPアドレス自動取得)
const char* ssid     = "XXXXX";
const char* password = "YYYYY";

// ウェブサーバーをポート80で開始
WiFiServer server(80);

// HTTPリクエストを保存しておく変数
String header;

// 温度計算
float temp_calc() {

  // 変数宣言
  float ONDO;                                       // 温度の計算結果
  unsigned int SENSOR_DATA;                         // 温度センサーから取り出した生データ(16ビット結合)
  unsigned int cmd[2];                              // I2Cで取り出したデータを格納する配列

  // 温度センサーに非ホールドマスターモードで測定トリガを送る
  Wire.beginTransmission(0x4A);                     // 0x4A -> センサーのアドレス(7ビット)
  Wire.write(0xF3);                                 // 非ホールドマスターモードの測定トリガ
  Wire.endTransmission(true);
  delay(100);                                       // 測定完了時間(＋α)待つ
  Wire.requestFrom(0x4A,2,true);                    // 2バイト読み込み(チェックサム省略)
  cmd[0] = Wire.read();
  cmd[1] = Wire.read();
  SENSOR_DATA = cmd[0]*256 + cmd[1];                // cmd[0]は上位8ビットなので256倍してから足し算
  ONDO = (float)SENSOR_DATA * 175.72 / 65536 - 46.85;  // 仕様書上の計算式に基づいて摂氏温度に換算

  // チェック用に幾つかシリアルモニタに出力
  Serial.print("1バイト目データ  ");
  Serial.println(cmd[0]);
  Serial.print("2バイト目データ  ");
  Serial.println(cmd[1]);
  Serial.print("温度計算結果  ");
  Serial.print(ONDO);
  Serial.println("(℃)");

  return ONDO;                                      // 計算結果をリターン値とする
}

void setup() {
  // シリアル通信ボーレート設定
  Serial.begin(115200);

  // I2Cポート割り当て
  Wire.begin(21, 22);                               // ピンは(SDA,SCL)の、順

  // ssidとpasswordを用いてWi-Fiに接続
  Serial.print("Connecting to ");
  Serial.println(ssid);
  WiFi.begin(ssid, password);
  while (WiFi.status() != WL_CONNECTED) {
    delay(500);
    Serial.print(".");
  }

  // 取得したIPアドレスをシリアルモニタに出力し、webserverをスタート
  Serial.println("");
  Serial.println("WiFi connected.");
  Serial.println("IP address: ");
  Serial.println(WiFi.localIP());
  server.begin();
}

void loop() {
  WiFiClient client = server.available();           // クライアント（スマホやPCなど）がつながっているかどうかをclientに出力
  float ondo;                                       // 温度測定結果

  if (client) {                                     // クライアントが来たとき
    Serial.println("New Client.");
    String currentLine = "";                        // クライアントからくるデータを格納する変数
    while (client.connected()) {                    // クライアントがつながっている間、以下をループ
      if (client.available()) {                     // クライアントからデータが来ているとき
```

```
        char c = client.read();                     // データを読み込み
        Serial.write(c);                            // 届いたデータをシリアルモニタに出力
        header += c;
        if (c == '¥n') {                            // 届いたデータが改行コードだった時
          // もし現在の行が空白ならば、この改行コードのみ受け取る
          // つまりHTTPリクエストの終わりなので、レスポンスを返す
          if (currentLine.length() == 0) {
            ondo = temp_calc();                     // 温度測定結果を代入
            // HTTPヘッダは (HTTP/1.1 200 OK)のようなステータスコードから始まる
            // 次にコンテントタイプを送信。今回はhtml形式なので以下のようにする
            client.println("HTTP/1.1 200 OK");
            client.println("Content-type:text/html; charset=utf-8;");
            client.println("Connection: close");
            client.println();

            // htmlを表示
            client.println("<!DOCTYPE html><html>");
            client.println("<head><meta name = ¥"viewport¥" content = ¥"width=device-width, initial-scale = 1¥">");
            client.println("<link rel = ¥"icon¥" href = ¥"data:,¥">");
            // 自動リフレッシュ
            client.println("<meta http-equiv = ¥"refresh¥" content = ¥"3¥"/>");
            // タイトル
            client.println("<title>ESP32_I2C_Test</title>");
            client.println("<style>h1{text-align: center;}");
            client.println("</style></head>");


            // ページ本体 (bodyタグ内)
            client.println("<body><h1>ESP32_I2C温度計お試し</h1>");
            client.println();
            client.println("<body><h1>現在の温度は");
            if (ondo > 30) {                          // 温度が30℃より高ければ赤で温度を表示
              client.println("<font color = ¥"red¥">");
              client.println(ondo, 2);
              client.println("</font>");
            }
            else if (ondo < 20) {                     // 温度が20℃より低ければ青で温度を表示
              client.println("<font color = ¥"blue¥">");
              client.println(ondo,2);
              client.println("</font>");
            }
            else {
              client.println(ondo, 2);
            }
            client.println("℃です</h1>");

            client.println("</body></html>");

            //  HTTPレスポンスの最後は改行で終了
            client.println();
            // whileループの終了
            break;
          }
          else {                                    // 改行コードを取得したら、currentLineをリセット
            currentLine = "";
          }
        }
        else if (c != '¥r') {                       // 改行以外の何かしらのコードが来ているとき
          currentLine += c;                         // currentLineに追加
        }
      }
    }
    // ヘッダーをリセット
    header = "";
    // 接続をリセット
    client.stop();
    Serial.println("Client disconnected.");
    Serial.println("");
  }
}
```