Arduino Mega
Temperatur, Uhrzeit und Datum im zweizeiligen Display

Volker Schuhmaier

2023

```

Arduino Mega mit zweizeiligem LCD-Display, RTC und DHT11.
Temperatur und Luftfeuchtigkeit werden in der erten Zeile dargestellt.
Uhrzeit und Datum, im Wechel, werden in der zweiten Zeile dargestellt.
Es werden zwei Automaten (Timer_DHT11 und Diplaychange) verwendet,
dadurch wird die Uhrzeit jede Sekunde aktualisiert.
Die Aktualisierung von Temperatur und Luftfeuchte erfolgt pro Sekunde;
der DHT11 benoetigt 1 Sekunde zum Auslesen.
*/


#include "RTClib.h"
#include <LiquidCrystal.h>
#include <SimpleDHT.h>

#define countof(a) (sizeof(a) / sizeof(a[0]))

char daysOfTheWeek[7][12] = { "Sunday", "Monday", "Tuesday", "Wednesday", "Thursday", "Friday", "Saturday" };
char daysOfTheWeek_DE[7][12] = { "Sonntag", "Montag", "Dienstag", "Mittwoch", "Donnerstag", "Freitag", "Samstag" };
char daysOfTheWeek_DE_kurz[7][12] = { "So", "Mo", "Di", "Mi", "Do", "Fr", "Sa" };

int pinDHT11 = 2; // Data Pin 2

// Generally, you should use "unsigned long" for variables that hold time
// The value will quickly become too large for an int to store
bool boolTimer_DHT11Read = 0;
bool boolDisplayChange = 0;
bool boolDisplayTime = 0;

byte byteTemperature = 0;
byte byteHumidity = 0;

int err = SimpleDHTErrSuccess;

const long longIntervalDHT11 = 500;    // (milliseconds) Interval fuer Anzeige des DHT11, allerdings dauert die Abfrage des DHT11 eine Sekunde
const long longDisplayChange = 5000;   // (milliseconds) Interval fuer Wechsel zwischen Uhrzeit und Datum
```

```

unsigned long ulongCurrMillis = millis(); //current Milliseconds
unsigned long ulongPrevMillisDHT11 = 0;   // previous Milliseconds for DHT read
unsigned long ulongPrevMillisDisplay = 0;  // previous Milliseconds for Display change

RTC_DS3231 rtc;

SimpleDHT11 dht11(pinDHT11);

LiquidCrystal lcd(7, 8, 9, 10, 11, 12);

void setup()
 {
    lcd.begin(16, 2);
    rtc.begin();
 }

void loop()
 {
  DateTime now = rtc.now();

  err = SimpleDHTErrSuccess;

  if ((err = dht11.read(&byteTemperature, &byteHumidity, NULL)) != SimpleDHTErrSuccess)
   {
    return;
   }

  ulongCurrMillis = millis();

  if (ulongCurrMillis - ulongPrevMillisDHT11 >= longIntervalDHT11)
   {
    ulongPrevMillisDHT11 = ulongCurrMillis;
    boolTimer_DHT11Read = 1;  // refresh time and humidity on Display
   }

  if (ulongCurrMillis - ulongPrevMillisDisplay >= longDisplayChange)
   {
    ulongPrevMillisDisplay = ulongCurrMillis;
    boolDisplayChange = 1;  // refresh Display with time or date
   }

  if (boolTimer_DHT11Read == 1)
   {
    lcd.setCursor(0, 0);
    lcd.print("T/H:    C /    %");
    lcd.setCursor(5, 0);
    lcd.print(byteTemperature);
    lcd.setCursor(11, 0);
    lcd.print(byteHumidity);

    boolTimer_DHT11Read = 0;  // Refresh done
   }

  // Display wird mit Uhrzeit und Datum getoggled
  if (boolDisplayChange == 1)
   {
     boolDisplayTime = !boolDisplayTime;

     boolDisplayChange = 0;
   }

  // Display-Auffrischung mit Datum
  if (boolDisplayTime == 0)
   {
    char charDatestring[20];
    snprintf_P(charDatestring,
               countof(charDatestring),
               PSTR("%02u.%02u.%04d"),
               now.day(),
               now.month(),
               now.year());

    lcd.setCursor(0, 1);
    lcd.print(daysOfTheWeek_DE_kurz[now.dayOfTheWeek()]);
    lcd.print(" ");
    lcd.print(charDatestring);
   }
  // Display-Auffrischug mit Uhrzeit
  if (boolDisplayTime == 1)
   {
    char charTimestring[20];
    snprintf_P(charTimestring,
               countof(charTimestring),
               PSTR("%02u:%02u:%02u"),
               now.hour(),
               now.minute(),
               now.second());

    lcd.setCursor(0, 1);
    lcd.print("                "); // Loesche den alten Inhalt
    lcd.setCursor(0, 1);
    lcd.print(charTimestring);
   }
}
```