```
// Source adapted from: https://myesp8266.blogspot.com/2017/12/bme280-and-esp8266.html
// FINAL TEST VERSION
// Copy and paste all text into the Arduino sketch editor

// Please include the following libararies via Arduino software!
#include <Wire.h>
#include <SPI.h>                //https://www.arduino.cc/en/Reference/SPI
#include <Adafruit_BME280.h>
#include <ESP8266WiFi.h>


Adafruit_BME280 bme; // I2C
unsigned long delayTime;

// Definition of the waiting time. Note this is a “blocking” code.
// Alternatively, please work with “millis”.

int delayTime_sec;

// replace with your channel’s thingspeak API key,
String apiKey = "YOUR-API-WRITE-KEY";      // insert the Key for write API from Thingspeak
const char* ssid = "YOUR SSID";       // Router
const char* password = "YOUR PASSWORD";    // Password
const char* server = "api.thingspeak.com";
WiFiClient client;


/**************************
 *   S E T U P
 **************************/
void setup() {
  Serial.begin(115200);   // Set the Baud rate
  Serial.println(F("BME280 measurement is started"));

// Change the waiting time here
  delayTime = 15000;

// for tests 15000 = 15sec or later depending on your application "n*60sec" .
// Thingspeak in free license version 15000 is the lowest value for data transmission.


  if (!bme.begin(0x76))
 {
  // 0x76 = set-up of IC2 adress = bits D7-D1, since I2C uses D0 for write/read.
  // connection from ESP8266 to BME280 sensor: 3.3V - VIN, G(bzw. 0) - GND, D1-SCL, D2-SDA
    Serial.println(F("Could not find a valid BME280 sensor, check wiring!"));
    while (1);
  }
  WiFi.begin(ssid, password);

  Serial.println();
  Serial.println();
  Serial.print(F("Connecting with WiFi: "));
  Serial.println(ssid);

  WiFi.begin(ssid, password);

  Serial.println(F("Connecting with WLAN: "));
  while (WiFi.status() != WL_CONNECTED)
  {
    delay(500);
    Serial.print(".");
  }
  Serial.println("");
  Serial.println(F("WiFi connected"));
  Serial.print(F("Bitte "));
  delayTime_sec = (delayTime / 1000);
  Serial.print(delayTime_sec);
  Serial.println(F(" s warten"));

}

  /**************************
 *  L O O P of main program cycle
 **************************/
```

```
void loop() {

        // declaration of  (TEMP, HUMI, PRESS) and correction factors
// Alternative: Use inside the Loop the lines   //postStr += String(bme.readxxx()); verwenden


        delay(delayTime);
        float TEMP = bme.readTemperature();
        float HUMI = bme.readHumidity();
        float PRESS = bme.readPressure();
        float PRESS_ROUND;
        float TEMP_ROUND;
        float HUMI_ROUND;
        float TempCor = 0.983;  // calibration Temperature
        float HumCor = 1.178; // calibration Humidity
        float PressCor = 1.02668;
       //calibration Baromectrical pressure. Use next station for reference.


    if (client.connect(server,80))  // "184.106.153.149" or api.thingspeak.com

 // Sending data to Thingspeak

    {
            String postStr = apiKey;

               postStr +="&field1=";
               TEMP_ROUND = round ((TEMP*TempCor)*10)/10.0;
               postStr += String(TEMP_ROUND);


               postStr +="&field2=";
               HUMI_ROUND = round ((HUMI*HumCor)*10)/10.0;
               postStr += String(HUMI_ROUND);


               PRESS_ROUND = round (((PRESS)*PressCor /100.0F)*10)/10.0;  // round to one dec
               postStr +="&field3=";
               postStr += String(PRESS_ROUND);

               postStr += "\r\n\r\n";


        client.print("POST /update HTTP/1.1\n");
        client.print("Host: api.thingspeak.com\n");
        client.print("Connection: close\n");
        client.print("X-THINGSPEAKAPIKEY: "+apiKey+"\n");
        client.print("Content-Type: application/x-www-form-urlencoded\n");
        client.print("Content-Length: ");
        client.print(postStr.length());
        client.print("\n\n");
        client.print(postStr);
        delay(1000);


  // These lines are only for showing in the Arduino Monitor
        Serial.println(F("Data sent to Thingspeak (API Key adaptation!!):"));
        Serial.print(F("actual API write key = ")); Serial.println(apiKey);
        Serial.println(F(" "));
        Serial.print(F("Temperature in Celsius  =     "));
        Serial.println(TEMP_ROUND);                          //auch hier anpassen
        Serial.print(F("Humidity in %  =          "));
        Serial.println(HUMI_ROUND);
        Serial.print(F("PRESS_ROUND to Thingspeak = "));
        Serial.println(PRESS_ROUND);
        Serial.println(F("Comparison of values for barometrical pressure:"));
        Serial.print(F("BME - read value of BME, hPa =   "));
        Serial.println(PRESS/100.0F);
        Serial.print(F("Adapted value in hPa - without round =          "));
        Serial.println((PRESS)*PressCor /100.0F);                //angepasst
        Serial.println(F(" "));
        Serial.print(F(" *** New Measurement in: "));
        Serial.print(delayTime_sec);
        Serial.println(F(" s"));
        Serial.println(F(" "));

  if (delayTime < 15000) {
        Serial.println(F("Warning: used cycle time 'delayTime' to short!"));
```

```
        Serial.println(F("Thingspeak free-license allows only data each 15000ms = 15s!"));
        Serial.println(F(" "));
        }
  else {
       Serial.print(F("Sending data to Thingspeak every: "));
       Serial.print(delayTime_sec);
       Serial.println(F(" s"));
       Serial.println(F(" "));
  }

    }

    client.stop();

    delay(200);

}  // End Loop
```