```
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
/*                     VERSAO RPi TESTE DO CODIGO - UPDATED 050118
               VERAO 3.0           DATA: 01072017
               COMPILADO NA VERSAO ARDUINO: 1.8.1
               _________________________________
                PLACA WIFI ESP8266-07 AT THINKER
                PROGRAMA: MINI ESTACAO CLIMATICA
                CONTÃ‰M SENSORES: BMP-180 E DHT22
               _________________________________
               CONFIGURACAO PLACA GRAVACAO - ESP-07
               FUNCIONA COM BIBLIOTECA ESP-07 COMMUNITY ATE VERSAO 2.3.0
               ATENCAO - VERSAO 2.4 NAO FUNCIONA PARA ESTE MODELO ESP-07
               _________________________________
               ATENCAO NAO COMPILAR ESP-07 NA VERSAO 2.4 OU SUPERIOR!!!!
               _________________________________
               PLACA:              GENERIC ESP8266 MODULE
               BIBLIOTECA          2.3.0
               FLASH MODE:         DIO
               FLASH SIZE:         1M (512K SPIFFS)
               DEBUG PORT:         DISABLED <--<
               DEBUG LEVEL:        TOUT <--<
               RESET MOTHOD:       ck
               FLASH FREQUENCY:    40 MHz
               CPU FREQUENCY:      80 MHz
               UPLOAD SPEED:       115200
               PORTA: PORTA ESP CONECTADA AO COMPUTADOR
               _________________________________
               CONFIGURACAO PLACA GRAVACAO - ESP-12E
               PLACA:              NODE MCU 1.0 (ESP-12E MODULE)
               CPU FREQUENCY:      80 MHz
               FLASH SIZE:         4M (1M SPIFFS)
               UPLOAD SPEED:       115200
               BIBLIOTECA          2.3.0 >--> +/-
               IwIP variant:       v.1.4 preBuilt >--> compila + erro flash ao gravar!
               IwIP variant:       v.1.4 preBuilt OpenSource >--> nao compila
               IwIP variant:       v.2 preBuilt (MSS=536) >--> compila mas nao funciona e ↲
               corrompe a memoria do ESP
               PORTA: PORTA ESP CONECTADA AO COMPUTADOR
               _________________________________                      */
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
#include <Wire.h>                // NECESSARIO COMUNICACAO I2C
#include <ESP8266WiFi.h>
#include <WiFiClient.h>
#include <ArduinoJson.h>
#include <stdlib.h>
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
// WiFi information
// const char WIFI_SSID[] = "cev2";
// const char WIFI_PSK[] = "TplnkAngelica2015";
//const char WIFI_SSID[] = "Mixceviana";   // celular
//const char WIFI_PSK[] = "0123456789";
//const char WIFI_SSID[] = "CEV_UNIFIQUE_2GHz";
//const char WIFI_PSK[] = "UnfqAngelica2015";
const char WIFI_SSID[] = "ESP_GUEST"; // visitante 2,4GHz TPLINK - 0k
const char WIFI_PSK[] = "01234567890";
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
String WIFI_IP = "NONE";
String WIFI_MAC = "NONE";
String SN="776543";
String VERSAO="v1.0r4";
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
// Remote site information
const char http_site[] = "www.api.app4iot.com.br";
const int http_port = 80; // https criptografia 443
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
// Pin definitions
const int LED_PIN = 13;     // nodeMCU
//const int LED_PIN = 12;     // atl board led1
//const int LED_PIN = 16;     // atl board led0
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
// CONSTANTS
// constants won't change. Used here to set a pin number:
const int ledPin =  LED_BUILTIN;// the number of the LED pin
// Generally, you should use "unsigned long" for variables that hold time
// The value will quickly become too large for an int to store
unsigned long previousMillis = 0;        // will store last time LED was updated
// constants won't change:
//const long interval = 60000;           // interval at which to blink (milliseconds)
//    intervalo = 300000;                  // 5 MINUTOS (TEMPO DE SUBIDA)
//    intervalo = 60000;                  // 1 MINUTO (TEMPO DE SUBIDA)
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
// Global variables
unsigned long   counterLocal = 0;
// Variables will change:
int ledState = LOW;             // ledState used to set the LED
WiFiClient client;
String _buffer;
char v1[10];
char v2[10];
char v3[10];
char v4[10];
char v5[10];
char cnt[10];
long l1,l2,l3,l4,l5,lcnt;
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
void setup() {
// Set up serial console to read web page
  Serial.begin(115200);
  Serial.print("Thing GET Example");
// Set up LED for debugging
  pinMode(LED_PIN, OUTPUT);
  digitalWrite(LED_PIN, LOW);
Serial.println("... 0 COMUTANDO ESTADO DO LED ...");
// Connect to WiFi
  connectWiFi();
WIFI_IP = WiFi.localIP().toString();
WIFI_MAC = WiFi.macAddress();
  Serial.print("\nIP LAN: ");
  Serial.println(WiFi.localIP());
  Serial.println("MAC: " + WiFi.macAddress());
// Attempt to connect to website
  if ( !getPage() ) {
    Serial.println("GET request failed");
  }
```

```
}
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
void loop() {
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
smartDelay(300000);   // 60000 = 1 min; 300000 = 5 min
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
// If there are incoming bytes, print them
  if ( client.available() ) {
    String c = client.readStringUntil('\r');
    //char c = client.read();
    //Serial.print(c);
    //Serial.println(" Contando");
    int state_pos = c.indexOf("{");
    if (state_pos > 0)
      _buffer = c;
  }
// If the server has disconnected, stop the client and WiFi
  if ( !client.connected() ) {
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
Serial.println("... 5 COMUTANDO ESTADO DO LED ...");
digitalWrite(LED_PIN, HIGH);
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
    Serial.print("Aqui------");
    Serial.println(_buffer);
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
// CONTADOR DE EVENTOS LOCAL - NAO SOBE PARA O SERVIDOR
Serial.print("Medida realizada: [ "); Serial.print(counterLocal);
Serial.println(" ]"); counterLocal++;
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
    Serial.println();
    StaticJsonBuffer<200> jsonBuffer;
    JsonObject& root = jsonBuffer.parseObject(_buffer);
    if ( root.success( )) {
      String _action = root["act"];
      boolean _status = root["status"];
      Serial.print("Acao : " );
      Serial.println(_action);
      Serial.print("Status: ");
      Serial.println(_status);
      if ( _action == "ON" ) {
        digitalWrite(LED_PIN, HIGH);
        Serial.println("... 2 COMUTANDO ESTADO DO LED ...");
      }
      if ( _action == "OFF") {
        digitalWrite(LED_PIN, LOW);
        Serial.println("... 3 COMUTANDO ESTADO DO LED ...");
      }
    }
    delay(5000);
Serial.println("... AGUARDOU DELAY DE 5000 ...");
// Attempt to connect to website
    if ( !getPage() ) {
      Serial.println("GET request failed");
    }
// Close socket and wait for disconnect from WiFi
    /*
    client.stop();
    if ( WiFi.status() != WL_DISCONNECTED ) {
      WiFi.disconnect();
    }
    // Turn off LED
    //digitalWrite(LED_PIN, LOW);
    // Do nothing
    Serial.println("Finished Thing GET test");
    while(true){
      delay(1000);
    }
    */
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
Serial.println("... 6 COMUTANDO ESTADO DO LED ...");
digitalWrite(LED_PIN, LOW);
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
}}
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
// Attempt to connect to WiFi
void connectWiFi() {
  byte led_status = 0;
// Set WiFi mode to station (client)
  WiFi.mode(WIFI_STA);
// Initiate connection with SSID and PSK
  WiFi.begin(WIFI_SSID, WIFI_PSK);
// Blink LED while we wait for WiFi connection
  while ( WiFi.status() != WL_CONNECTED ) {
//digitalWrite(LED_PIN, led_status);
//led_status ^= 0x01;
    Serial.print(".");
    delay(100);}
// Turn LED on when we are connected
  digitalWrite(LED_PIN, LOW);
  Serial.println("... 4 COMUTANDO ESTADO DO LED ...");}
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
// Perform an HTTP GET request to a remote page
bool getPage() {
// Attempt to make a connection to the remote server
  Serial.print(" Site http: ");
  Serial.print(http_site);
  Serial.print(" Porta: ");
  Serial.println(http_port);
  if ( !client.connect(http_site, http_port) ) {
    return false;}
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
  l1 = random(100);
  l2 = random(200);
  l3 = random(300);
  l4 = random(400);
  l5 = random(500);
  lcnt = random(600);
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
  ltoa(l1, v1, 10);
  ltoa(l2, v2, 10);
  ltoa(l3, v3, 10);
  ltoa(l4, v4, 10);
  ltoa(l5, v5, 10);
  ltoa(lcnt, cnt, 10);
```

```
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
String _comando;
  _comando  = "GET
  /api-app4iot.php?act=GRV&tkn=iop2oi1jhu87hgn1hgqnbaiuk187jd&dis=cab4lpawluyxc30pkhb1mw410k1
  7x9&var=wlg7tvk6wmhr91hc08q4wh2ik7mqa7";
  _comando += "&v1=";
  _comando += v1;
  _comando += "&v2=";
  _comando += v2;
  _comando += "&v3=";
  _comando += v3;
  _comando += "&v4=";
  _comando += v4;
  _comando += "&v5=";
  _comando += v5;
  _comando += "&cnt=";
  _comando += cnt;
  _comando += "&ip=";
  _comando += WIFI_IP;
  _comando += "&mac=";
  _comando += WIFI_MAC;
  _comando += "&sid=";
  _comando += WIFI_SSID;
  _comando += "&pwd=";
  _comando += encrypty(WIFI_PSK);
  _comando += "&sn=";
  _comando += SN;
  _comando += "&ver=";
  _comando += VERSAO;
  _comando += " HTTP/1.1";
Serial.println(_comando);  // faz o print na serial da pagina web inteira...
// Make an HTTP GET request
  client.println(_comando);
  client.print("Host: ");
  client.println(http_site);
  client.println("Connection: close");
  client.println();
return true;}
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
String encrypty(String valor) {
  String k1 = "1234567890abcdefghijklmnopqrstuvxz!@#$%*()_+-=[{]};:.>,<?";
  String k2 = "?<,>;.:}]{[=-+_)(*%$#@!zxvutsrqponmlkjihgfedcea0987654321";
  int tam = valor.length();
  String pos = "";
  int pos_k1;
  char pos_k2;
  for (int p=0; p<tam; p++ ) {
      pos = valor.charAt(p);
      pos_k1 = k1.indexOf(pos);
      pos_k2 = k2.charAt(pos_k1);
      valor.setCharAt( p, pos_k2);;}
return valor;}
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
void smartDelay(unsigned long interval){
  unsigned long currentMillis = millis();
  if (currentMillis - previousMillis >= interval) {
// save the last time you blinked the LED
```

```
    previousMillis = currentMillis;
  Serial.print("... NA FUNCAO smartDelay ... t = ");
  Serial.println(interval);
// if the LED is off turn it on and vice-versa:
//    if (ledState == LOW) {
//      ledState = HIGH;} else {ledState = LOW;}
// set the LED with the ledState of the variable:
}}
/* - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - - -*/
```