NEXTION

Variables globales

int sys0=0,sys1=0,sys2=0

int templen=0,templendec=0,temp=0,temp2=0,tempM=0,tempH=0

int TaraHecha=0

int float=0

int ultELEMENTO=0, ultPAGINA=0

int temp\_prog1,temp\_prog2,temp\_prog3,temp\_prog4,temp\_prog5

int timer=0,SERIAL\_DEBUG,timer\_beep,vent\_auto\_off //"bool"

int set\_time\_min

int VentMinDutyCycle

int calibration\_factor1,calibration\_factor1\_ENT,calibration\_factor1\_DEC

int zero\_offset1,zero\_offset2

int tara1,tara2

int bascula1\_calibrada,bascula2\_calibrada

int calibration\_factor2,calibration\_factor2\_ENT,calibration\_factor2\_DEC

int peso\_calibrado1,peso\_calibrado2

int vaggKp, vaggKp\_ENT, vaggKp\_DEC

int vaggKi, vaggKi\_ENT, vaggKi\_DEC

int vaggKd, vaggKd\_ENT, vaggKd\_DEC

int aggKp, aggKp\_ENT, aggKp\_DEC

int aggKi, aggKi\_ENT, aggKi\_DEC

int aggKd, aggKd\_ENT, aggKd\_DEC

int consKp, consKp\_ENT, consKp\_DEC

int consKi, consKi\_ENT, consKi\_DEC

int consKd, consKd\_ENT, consKd\_DEC

int WindowSize

int dht\_T\_ext\_correccion, dht\_T\_ext\_correccion\_ENT

int dht\_T\_int\_correccion, dht\_T\_int\_correccion\_ENT

int dht\_H\_ext\_correccion, dht\_H\_int\_correccion

int time\_prog1,time\_prog2,time\_prog3,time\_prog4,time\_prog5

int WeightReadingRefreshTime

int setpoint

int neopixel\_pixels

int ledStatus

int peso\_tara1,peso\_tara2,peso\_tara3,peso\_tara4,peso\_tara5

int pesoConocidoCAL1,pesoConocidoCAL2

int pesoAjustadoCAL1,pesoAjustadoCAL2

int dht\_T\_ext,dht\_T\_int

int dht\_H\_ext,dht\_H\_int

int PesoInicial1,PesoInicial2

int PESO1, PESO2

int hInicial, mInicial

int timeSTART, timeLEFT, timeFINAL, timeACT, timePAST

int medianoche

string main.name\_prog1, main.name\_prog2, main.name\_prog3, main.name\_prog4, main.name\_prog5

string main.name\_tara1, main.name\_tara2, main.name\_tara3, main.name\_tara4, main.name\_tara5

string neopixel\_type

string teclado.INPUTstring

int teclado.INPUT

int numpad.coma //si se activa la coma decimal o no (para números enteros)

ARDUINO

Settings EEPROM

bool SERIAL\_DEBUG;

char name\_prog1[6];

char name\_prog2[6];

char name\_prog3[6];

char name\_prog4[6];

char name\_prog5[6];

byte temp\_prog1;

byte temp\_prog2;

byte temp\_prog3;

byte temp\_prog4;

byte temp\_prog5;

bool timer;

unsigned int set\_time\_min;

bool timer\_beep;

bool vent\_auto\_off;

byte VentMinDutyCycle;

float calibration\_factor1;

long zero\_offset1;

bool bascula1\_calibrada;

float calibration\_factor2;

long zero\_offset2;

bool bascula2\_calibrada;

float vaggKp;

float vaggKi;

float vaggKd;

float aggKp;

float aggKi;

float aggKd;

float consKp;

float consKi;

float consKd;

unsigned int WindowSize;

float dht\_T\_ext\_correccion;

float dht\_T\_int\_correccion;

float dht\_H\_ext\_correccion;

float dht\_H\_int\_correccion;

unsigned int time\_prog1;

unsigned int time\_prog2;

unsigned int time\_prog3;

unsigned int time\_prog4;

unsigned int time\_prog5;

unsigned long WeightReadingRefreshTime;

float setpoint;

int neopixel\_type;

int neopixel\_pixels;

bool ledStatus;

char name\_tara1[10];

char name\_tara2[10];

char name\_tara3[10];

char name\_tara4[10];

char name\_tara5[10];

int peso\_tara1;

int peso\_tara2;

int peso\_tara3;

int peso\_tara4;

int peso\_tara5;

int ultima\_tara1;

int ultima\_tara2;

char ver[sizeof(SETTING\_VER)];

Variables Arduino

#define NEOPIXEL\_PIN 3 //interrupt, pwm capable

#define ALARMA 4

#define RELE\_RESISTENCIA 5 //pwm capable

#define RELE\_VENTILADOR 6 //pwm capable

#define PESO\_DATA2 8

#define PESO\_SCK2 9 //pwm capable

#define PESO\_SCK 10 //pwm capable, SPI-SS

#define PESO\_DATA 11 //pwm capable, SPI-MOSI

#define SWITCH 12 //SPI MISO

#define SENSOR\_INT A2

#define SENSOR\_EXT A3

//neopixel

#define NEOPIXEL\_LED //comentar si no hay tira LED

#define NEOPIXEL\_IS\_SEQUENTIAL

#define NEOPIXEL\_BRIGHTNESS 127

#define NEOPIXEL\_STARTUP\_TEST

#define NEOPIXEL\_BKGD\_LED\_INDEX

#define NEOPIXEL\_BKGD\_COLOR { 255, 255, 255, 0 } // R, G, B, W

int NEOPIXEL\_PIXELS;

bool ledStatus;

bool ledFlip=false;

bool LEDHeatingDONE=false;

unsigned long ledTime;

unsigned long ledTimeInterval = 300; //Tiempo de cambio de los LED ms

uint32\_t colorOff = neo->Color( 0, 0, 0);

uint32\_t colorRed = neo->Color(255, 0, 0);

uint32\_t colorOrange = neo->Color(255, 80, 0);

uint32\_t colorYellow = neo->Color(255, 255, 0);

uint32\_t colorGreen = neo->Color( 0, 255, 0);

uint32\_t colorBlue = neo->Color( 0, 0, 255);

uint32\_t colorIndigo = neo->Color( 0, 255, 255);

uint32\_t colorViolet = neo->Color(255, 0, 255);

uint32\_t colorWhite = neo->Color(255, 255, 255);

uint16\_t hue = 65536/3; //Color Verde

//serial

unsigned long serialTime;

char received; //para la funcion SerialReceive

// ARDUINO RTC CLOCK

int hora\_actual;

int minuto\_actual;

// PID

float input, output;

float setpoint;

//Define the very aggressive, aggressive and conservative Tuning Parameters

float vaggKp, vaggKi, vaggKd;

float aggKp, aggKi, aggKd;

float consKp, consKi, consKd;

unsigned int WindowSize;

unsigned long windowStartTime;

bool HeatingDONE = false;

// BASCULAS Y PESOS

#define RESOLUTION 1 //decimal places of HX711 reading TO BE SHOWN (actual reading is float)

#define AVG\_FACT 30 //how many reads for averaging factor of HX711

unsigned long WeightReadingRefreshTime; //ms entre lecturas de peso y temp. exterior (minimo 2seg)

unsigned long LastWeightTime = 0;

//bascula 1

float calibration\_factor1;

long zero\_offset1;

long tara1\_actual\_g;

long tara1\_actual\_offset;

volatile float PESO1 = 0.0f;

bool bascula1\_conectada = true; //por defecto conectada

bool tara1\_cambio = true; //cuando cuambias de tara (rollo de filamento distinto) cambiar a false para produccion

bool bascula1\_calibrada = false;

//bascula 2

float calibration\_factor2;

long zero\_offset2;

long tara2\_actual\_g;

long tara2\_actual\_offset;

volatile float PESO2 = 0.0f;

bool bascula2\_conectada = true; //por defecto conectada

bool tara2\_cambio = false; //cuando cuambias de tara (rollo de filamento distinto)

bool bascula2\_calibrada = false;

int peso\_tara1;

int peso\_tara2;

int peso\_tara3;

int peso\_tara4;

int peso\_tara5;

int ultima\_tara1;

int ultima\_tara2;

// SENSORES DE TEMPERATURA Y HUMEDAD

//sensor exterior

float dht\_T\_ext\_correccion;

float dht\_H\_ext\_correccion;

float T\_ext;

float H\_ext;

//sensor interior

float dht\_T\_int\_correccion; //el sensor AM2301 es mas preciso, el otro se corrige

float dht\_H\_int\_correccion; //el sensor AM2301 es mas preciso, el otro se corrige

float T\_int;

float H\_int;

float start\_temp;

float target\_temp;

float current\_temp;

unsigned long tempTime;

unsigned long tempTimeInterval = 4000; //Tiempo entre lecturas del sensor de temperatura interior

//temperaturas programadas

byte temp\_prog1;

byte temp\_prog2;

byte temp\_prog3;

byte temp\_prog4;

byte temp\_prog5;

// VENTILADOR

bool vent\_auto\_off;

byte VentMinDutyCycle; //cuando el dutyCycle de la resistencia es menor que este numero el ventilador se apaga

// TEMPORIZADOR Y TIEMPOS

// ------------------------------------------------

unsigned int set\_time\_min; //Temporizador en minutos

bool timer; //activar/desactivar temporizador

bool timer\_beep; //señal acustica de fin de programa - temporizador

unsigned long now = 0; //ms

unsigned long startTime = 0; //ms

unsigned long totalTime = 0; //ms

bool programa\_terminado = false; //Para saber si el programa /temporizador ha terminado

bool empezar\_programa = false; //Cuando empieza el temporizador para iniciar variables iniciales

//temporizadores programados

unsigned int time\_prog1;

unsigned int time\_prog2;

unsigned int time\_prog3;

unsigned int time\_prog4;

unsigned int time\_prog5;

SETUP

//nombre de los programas;

String name\_prog1;

String name\_prog2;

String name\_prog3;

String name\_prog4;

String name\_prog5;

String name\_tara1;

String name\_tara2;

String name\_tara3;

String name\_tara4;

String name\_tara5;

int NEOPIXEL\_TYPE = settings.parameters->neopixel\_type;

NEOPIXEL\_PIXELS = settings.parameters->neopixel\_pixels;

ledStatus = settings.parameters->ledStatus;

// LEER PARÁMETROS DE LA CONFIGURACIÓN GUARDADA

// ------------------------------------------------

SERIAL\_DEBUG = settings.parameters->SERIAL\_DEBUG;

name\_prog1 = settings.parameters->name\_prog1;

name\_prog2 = settings.parameters->name\_prog2;

name\_prog3 = settings.parameters->name\_prog3;

name\_prog4 = settings.parameters->name\_prog4;

name\_prog5 = settings.parameters->name\_prog5;

temp\_prog1 = settings.parameters->temp\_prog1;

temp\_prog2 = settings.parameters->temp\_prog2;

temp\_prog3 = settings.parameters->temp\_prog3;

temp\_prog4 = settings.parameters->temp\_prog4;

temp\_prog5 = settings.parameters->temp\_prog5;

timer = settings.parameters->timer;

set\_time\_min = settings.parameters->set\_time\_min;

timer\_beep = settings.parameters->timer\_beep;

vent\_auto\_off = settings.parameters->vent\_auto\_off;

VentMinDutyCycle = settings.parameters->VentMinDutyCycle;

calibration\_factor1 = settings.parameters->calibration\_factor1;

zero\_offset1 = settings.parameters->zero\_offset1;

bascula1\_calibrada = settings.parameters->bascula1\_calibrada;

calibration\_factor2 = settings.parameters->calibration\_factor2;

zero\_offset2 = settings.parameters->zero\_offset2;

bascula2\_calibrada = settings.parameters->bascula2\_calibrada;

vaggKp = settings.parameters->vaggKp;

vaggKi = settings.parameters->vaggKi;

vaggKd = settings.parameters->vaggKd;

aggKp = settings.parameters->aggKp;

aggKi = settings.parameters->aggKi;

aggKd = settings.parameters->aggKd;

consKp = settings.parameters->consKp;

consKi = settings.parameters->consKi;

consKd = settings.parameters->consKd;

WindowSize = settings.parameters->WindowSize;

dht\_T\_ext\_correccion = settings.parameters->dht\_T\_ext\_correccion;

dht\_T\_int\_correccion = settings.parameters->dht\_T\_int\_correccion;

dht\_H\_ext\_correccion = settings.parameters->dht\_H\_ext\_correccion;

dht\_H\_int\_correccion = settings.parameters->dht\_H\_int\_correccion;

time\_prog1 = settings.parameters->time\_prog1;

time\_prog2 = settings.parameters->time\_prog2;

time\_prog3 = settings.parameters->time\_prog3;

time\_prog4 = settings.parameters->time\_prog4;

time\_prog5 = settings.parameters->time\_prog5;

WeightReadingRefreshTime = settings.parameters->WeightReadingRefreshTime;

setpoint = settings.parameters->setpoint;

name\_tara1 = settings.parameters->name\_tara1;

name\_tara2 = settings.parameters->name\_tara2;

name\_tara3 = settings.parameters->name\_tara3;

name\_tara4 = settings.parameters->name\_tara4;

name\_tara5 = settings.parameters->name\_tara5;

peso\_tara1 = settings.parameters->peso\_tara1;

peso\_tara2 = settings.parameters->peso\_tara2;

peso\_tara3 = settings.parameters->peso\_tara3;

peso\_tara4 = settings.parameters->peso\_tara4;

peso\_tara5 = settings.parameters->peso\_tara5;

ultima\_tara1 = settings.parameters->ultima\_tara1;

ultima\_tara2 = settings.parameters->ultima\_tara2;

LOOP

// LEER TIEMPO Y COMPROBAR TEMPORIZADOR

// ------------------------------------------------

now = millis(); //Iniciar variable now en cada bucle

bool temporizador\_end = false;

unsigned long CurrentWindow = now - windowStartTime;

float gap = abs(setpoint - input); //distance away from setpoint

// AUTO APAGADO DEL VENTILADOR

unsigned int dutyCycle = (output / WindowSize)\*100;