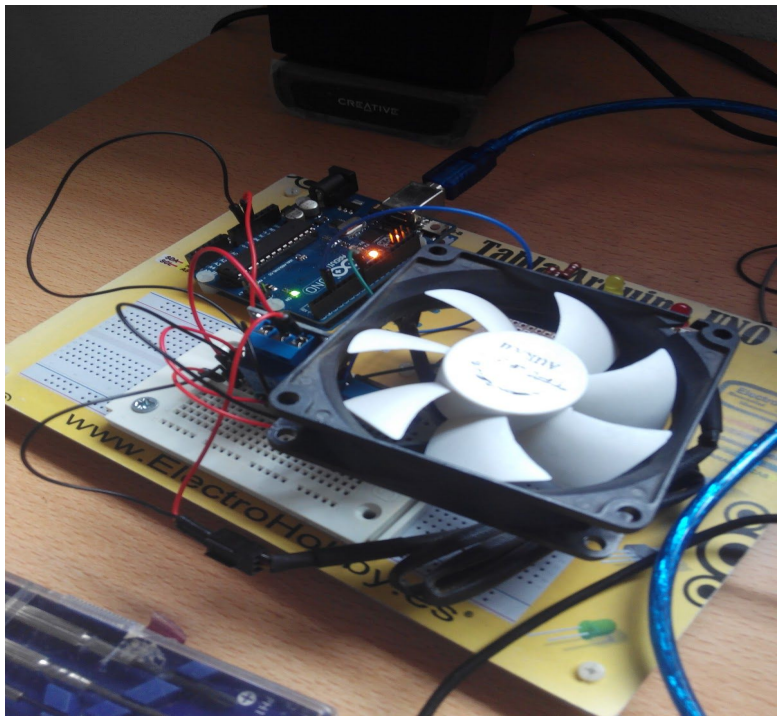
INTERNET DE LAS COSAS



PRÁCTICA – MONTAJE DEL DISPOSITIVO ARDUINO Y CONFIGURACIÓN CON LA PLATAFORMA THINGER.IO

MARIO LOSANTOS ALBACETE 04847925-P
mla4@um.es
MOISÉS FRUTOS PLAZA 48488132-S
moises.frutos@um.es

MÁSTER EN BIG DATA

ENTREGADO EL 4 DE ABRIL DE 2017

Índice

Primera parte: Arduino Ethernet

Introducción:

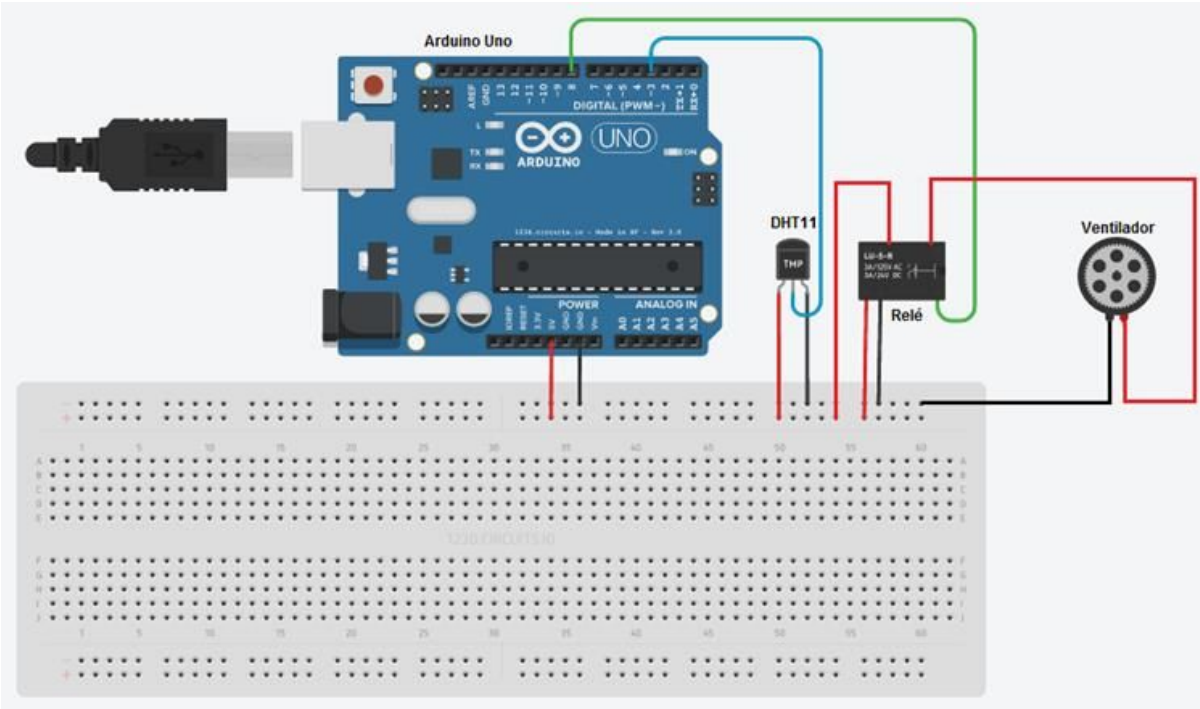
El objetivo de la práctica es el de familiarizarse con el entorno de desarrollo de la plataforma arduino y en el montaje de un sistema para la medición de la temperatura y la activación, condicionada a un aumento de la temperatura de un ventilador, controlado mediante un relé (o relay en inglés).

El sistema a componer está compuesto consiste en un sencillo esquema compuesto por:

- un microcontrolador arduino
- un protoboard o placa de interconexiones para facilitar el montaje
- un sensor de temperatura y humedad, de la familia *DHT11*
- un relé
- un ventilador
- un conjunto de conectores y un cable USB para programar la placa arduino

Esquema:

La primera parte de la práctica ha consistido en el montaje de los distintos componentes siguiendo el esquema proporcionado.



En él podemos observar que el sensor *DHT11* es conectado al PIN digital 3, que posteriormente será configurado como pin de entrada. El relé (interconectado como Normalmente Abierto) es controlado usando el PIN digital 8, que será configurado como pin de salida. El ventilador es conectado a la salida del relé para que cierre el circuito cuando se lo indique ese PIN 8. También falta por indicar que la salida de 5vcc es interconectada al sensor y al relé (a este tanto para alimentación como para conmute energía hacia el ventilador. El polo ground se usa para el relé, el ventilador y el sensor.

Configuración del entorno de desarrollo Arduino:

En este apartado tan sólo queremos destacar un par de requisitos para poder ejecutar el código una vez ha sido cargado en el entorno:

- es necesario indicar qué placa estamos utilizando para ello hay que acceder al menú Herramientas -> Placa -> *Arduino Ethernet* (o el que corresponda, *Arduino UNO R3* en nuestro caso).
- hay que enlazar la librería del sensor *DHT11*, para ello pulsamos en el menú Incluir librería -> *DHT11*.

Código:

El código resultado de la programación es el siguiente:

```
/*
 *  Moisés Frutos Plaza
 *  Mario Losantos Albacete
 */


#include "DHT.h"

#define PIN_SENSOR 3
#define PIN_RELAY 8
#define LED_INTERNO 13

#define DHTTYPE DHT11
#define UMBRAL_TEMPERATURA 1
#define UMBRAL_HUMEDAD 1

unsigned long last_relay_change = 0;
unsigned long t_delay = 1000;
unsigned long tiempo = 0;
float temperatura_inicial = 0;
float humedad_inicial = 0;

int relaystatus = HIGH;
int ledstatus = HIGH;
DHT dht(PIN_SENSOR, DHTTYPE);


void abrir_relay(){
  Serial.println("abrir_relay");
  if (relaystatus == HIGH){
```

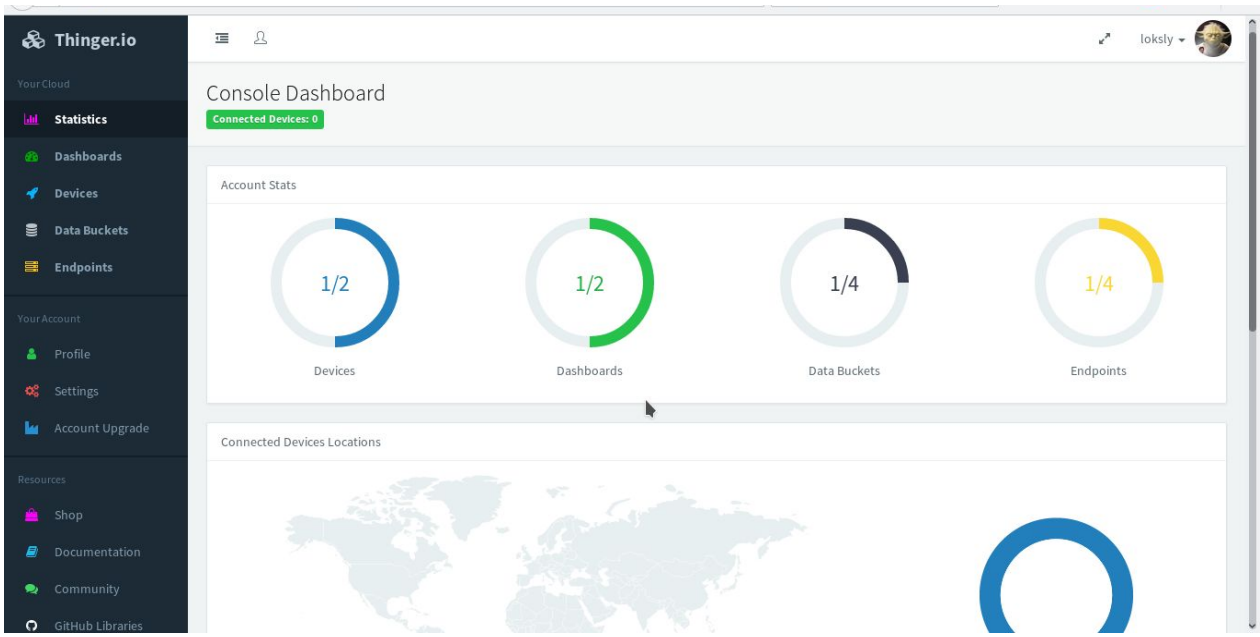
```
        digitalWrite(PIN_RELAY, LOW);
        relaystatus = LOW;
  }
}

void cerrar_relay(){
  Serial.println("cerrar_relay");
  if (relaystatus == LOW){
        digitalWrite(PIN_RELAY, HIGH);
        relaystatus = HIGH;
  }
}

float readTemperature(){
  float t = dht.readTemperature(false);
  //Serial.print("Temperatura:");
  //Serial.println(t);

  if (isnan(t)) {
        Serial.println("Failed to read from DHT sensor!");
        return 0;
  }
  return t;
}

float readHumidity(){
  float h = dht.readHumidity();
  //Serial.print("Humedad:");
  //Serial.println(h);

  if (isnan(h)) {
        Serial.println("Failed to read from DHT sensor!");
        return 0;
  }
  return h;
}

void ledOn(){
  if(ledstatus == LOW) {
        Serial.println("Encender Led");
        digitalWrite(LED_INTERNO, HIGH);
        ledstatus = HIGH;
  }
}

void ledOff(){
  if(ledstatus == HIGH) {
        Serial.println("Apagar Led");
        digitalWrite(LED_INTERNO, LOW);
        ledstatus = LOW;
  }
}

void setup() {
  Serial.begin(9600);
  Serial.println("Bienvenido a las practicas de IoT!");

  pinMode(PIN_SENSOR, INPUT);
  pinMode(PIN_RELAY, OUTPUT);
  pinMode(LED_INTERNO, OUTPUT);
  temperatura_inicial = readTemperature();
  //Serial.print("Temperatura Inicial: ");
  //Serial.println(temperatura_inicial);
  humedad_inicial = readHumidity();
  //Serial.print("Humedad Inicial: ");
  //Serial.println(humedad_inicial);
  ledOff();
  abrir_relay();
}
```

```
void check_temperature(){

 float temperatura = readTemperature();
  if (temperatura > (temperatura_inicial + UMBRAL_TEMPERATURA)){
        Serial.println("demasiado calor");
        cerrar_relay();
  } else if (temperatura < (temperatura_inicial - UMBRAL_TEMPERATURA)){
        Serial.println("demasiado refrigerado");
        abrir_relay();
  }
}

void check_humidity(){

 float humidity = readHumidity();
  if (humidity > (humedad_inicial + UMBRAL_HUMEDAD)){
        Serial.println("demasiada humedad");
        ledOn();
  } else if (humidity < (humedad_inicial - UMBRAL_HUMEDAD)){
        Serial.println("poca humedad");
        ledOff();
  }
}

void print_status() {
  Serial.print("LOW=");
  Serial.print(LOW);
  Serial.print(" HIGH=");
  Serial.print(HIGH);
  Serial.print(" Estado Relay: ");
  Serial.print(relaystatus);
  Serial.print("  Estado Led: ");
  Serial.print(ledstatus);
  Serial.print("  T. Inicial: ");
  Serial.print(temperatura_inicial);
  Serial.print("  T. Actual: ");
  Serial.print(readTemperature());
  Serial.print("  H. Inicial: ");
  Serial.print(humedad_inicial);
  Serial.print("  H. Actual: ");
  Serial.println(readHumidity());
}

void loop() {
  tiempo = millis();

  //delay artificial
  if( tiempo > last_relay_change + t_delay) {
        last_relay_change = tiempo;
        check_temperature();
        check_humidity();
        print_status();
  }
}
```

Segunda parte: Integración con la plataforma Thinger.io

Introducción:

El objetivo de esta segunda parte de la práctica es integrar nuestro dispositivo montado en la primera parte con la plataforma para IoT llamada *thinger.io*. Dentro de esa plataforma podremos observar en tiempo real el comportamiento del sensor de temperatura y el de humedad, así como activar o desactivar el relé del ventilador a voluntad.



Desarrollo:

Para llevar a cabo el desarrollo de esta parte de la práctica hemos seguido los siguientes pasos:

1. Darnos de alta en la plataforma https://thinger.io/ con la cuenta con login loksly@gmail.com.
2. Una vez dados de alta en la plataforma, agregamos nuestro dispositivo Arduino Uno R3 y generamos las credenciales.
3. A continuación descomprimimos las librerías para Arduino dentro de la subcarpeta *libraries*.
4. Registramos el dispositivo con las credenciales obtenidas en la plataforma (apartado *Devices*), y configuraremos la red (DHCP) utilizando la dirección MAC, que aparece

en la pegatina del propio Arduino, e introducimos el código relativo a la obtención de conectividad e identificación del dispositivo.

5. Declaramos los 3 elementos con los que trabajamos (temperatura, humedad y relé) como “cosas” de cara a la plataforma, teniendo en cuenta que dos elementos (temperatura y humedad) son variables de salida y el relé una variable de entrada (por tanto su declaración cambia).
6. Tras una ejecución exitosa de la aplicación procedemos a la generación de un panel de control (*dashboard*), al que vamos añadiendo widgets, entre ellos destaca un *Text/Value* para mostrar la temperatura o la humedad y un *device control* para activar y desactivar el relé. Empleamos un intervalo de muestreo (“sampling interval”) de 10 segundos para no sobrecargar a la plataforma de tráfico innecesariamente.
7. Una vez que el dispositivo ya nos aparecía como conectado en el gestor de dispositivos de thinger, creamos un “cuadro de mandos” o *Dashboard* para monitorizar la temperatura y la humedad y un interruptor para el relé. Al declarar cada uno de los widgets,
8. Una vez comprobamos que los widgets recibían los datos, añadimos también un “cubo de datos” o *Data Bucket* para almacenar los valores de temperatura y humedad.
9. Registramos otro panel de control que se alimentaba desde ese bucket, con un widget de tipo *Time Series Chart*. En él es posible ver la evolución de los datos proporcionados por el sensor.
10. A continuación creamos un “punto final” o *Endpoint* de tipo e-mail, e incluimos en el código de Arduino un aviso para que cuando la temperatura supere el umbral de 27 ºC definida en la constante MAX_TEMP. Hay que destacar que las notificaciones no se gestionan desde la plataforma *thinger* sino desde el dispositivo. es decir, el disparo del evento se hace en Arduino y hay que evitar que mande e-mails de manera masiva.
11. Comprobamos que en el ordenador todo funciona correctamente.
12. Tras ello descargamos la app para Android en el móvil. Escaneamos el código *QR* y de este modo ya nos aparecía el cuadro de mandos. Probamos a activar y desactivar el relé varias veces con éxito, aparte de visualizar la temperatura y la humedad.

Código:

A continuación incluimos el código definitivo para su evaluación:

```
#include <DHT.h>
#include <SPI.h>
#include <Ethernet.h>
#include <ThingerEthernet.h>

#include "DHT.h"

/*
```

```
 *  Moisés Frutos Plaza
 *  Mario Losantos Albacete
 */


/* Thinger.io vars */
#define USERNAME "loksly"
#define DEVICE_ID "dispositivoUMU"
#define DEVICE_CREDENTIAL "415ab40ae9b7cc4e66d6769cb2c08106e8293b48"
#define SENSORS "SENSORS"
#define TEMPERATURE_SENSOR "TEMPERATURE_SENSOR"
#define HUMIDITY_SENSOR "HUMIDITY_SENSOR"
#define RELAY "RELAY"

#define ENDPOINTEMAIL "endpointtest"


/* Local vars*/
#define PIN_SENSOR 3
#define PIN_RELAY 8

#define DHTTYPE DHT11
#define MAX_TEMP 27
#define MIN_TEMP 10
#define UMBRAL 3

ThingerEthernet thing(USERNAME, DEVICE_ID, DEVICE_CREDENTIAL);

int relaystatus = 1;
DHT dht(PIN_SENSOR, DHTTYPE);

unsigned long last_endpointcall = 0;
unsigned long last_relay_change = 0;
unsigned long t_delay = 5000;
unsigned long t_delay_endpoint = 60000;
unsigned long tiempo = 0;

bool connected_ = false;

bool connect_network(){
    if(connected_) return true;
    byte mac[] = { 0x90, 0xA2, 0xDA, 0x10, 0x81, 0x16 };
    unsigned long ethernet_timeout = millis();
    THINGER_DEBUG("NETWORK", "Initializing Ethernet...");
    while(Ethernet.begin(mac)==0){
        THINGER_DEBUG("NETWORK", "Getting IP Address...");
        if(millis() - ethernet_timeout > 30000) {
            delay(1000);
            return false;
        }
    }
    THINGER_DEBUG_VALUE("NETWORK", "Got IP Address: ", Ethernet.localIP());

    Serial.println(Ethernet.localIP());
    delay(1000);
    connected_ = true;
    return true;
}

void abrir_relay(){
  Serial.println("abrir_relay");
  if (relaystatus > 0){
    digitalWrite(PIN_RELAY, LOW);
    relaystatus = -1;
  }
}

void cerrar_relay(){
  Serial.println("cerrar_relay");
  if (relaystatus < 0){
```

```
    digitalWrite(PIN_RELAY, HIGH);
    relaystatus = 1;
  }
}

float readHumidity(){
  float t = dht.readHumidity();
  Serial.print("Humedad: ");
  Serial.println(t);

  if (isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");

    return 0;
  }

  return t;
}

float readTemperature(){
  float t = dht.readTemperature(false);
  Serial.print("Temperatura: ");
  Serial.println(t);

  if (isnan(t)) {
    Serial.println("Failed to read from DHT sensor!");

    return 0;
  }

  return t;
}

void sensorupdate(pson& out){
    out[TEMPERATURE_SENSOR] = readTemperature();
    out[HUMIDITY_SENSOR]  = readHumidity();
}


void setup() {
  bool connected = false;
  Serial.begin(9600);
  Serial.println("Programa iniciado");

  pinMode(PIN_SENSOR, INPUT);
  pinMode(PIN_RELAY, OUTPUT);
  dht.begin();

  do{
    if (connect_network()){
      thing[RELAY] << digitalPin(PIN_RELAY);
      thing[SENSORS] >> sensorupdate;
      connected = true;
      Serial.println("Tengo conexión");
    }
  }while(!connected);

}

void check_temperature(){
  float temp = readTemperature();
  if (temp > MAX_TEMP){
    callEndoint();
    Serial.println("demasiado calor");
    cerrar_relay();
  }else if (temp < MIN_TEMP){
    Serial.println("demasiado refrigerado");
    abrir_relay();
  }
}
```

```
void callEndoint(){
  tiempo = millis();
  if (tiempo > last_endpointcall + t_delay_endpoint){
    Serial.print("Voy a avisar a ");
    Serial.println(ENDPOINTEMAIL);
    thing.call_endpoint(ENDPOINTEMAIL);
    last_endpointcall = tiempo;
  }
}


void loop() {

  thing.handle();
  tiempo = millis();

  //delay artificial
  if( tiempo > last_relay_change + t_delay) {
    last_relay_change = tiempo;
    check_temperature();

    /*
    // envío explícito, innecesario una vez que se ha hecho la vinculación,
    // mediante thing[SENSORS] >> sensorupdate;

    thing[TEMPERATURE_SENSOR] >> outputValue(readTemperature());
    thing[HUMIDITY_SENSOR] >> outputValue(readHumidity());
    */
  }
}
```