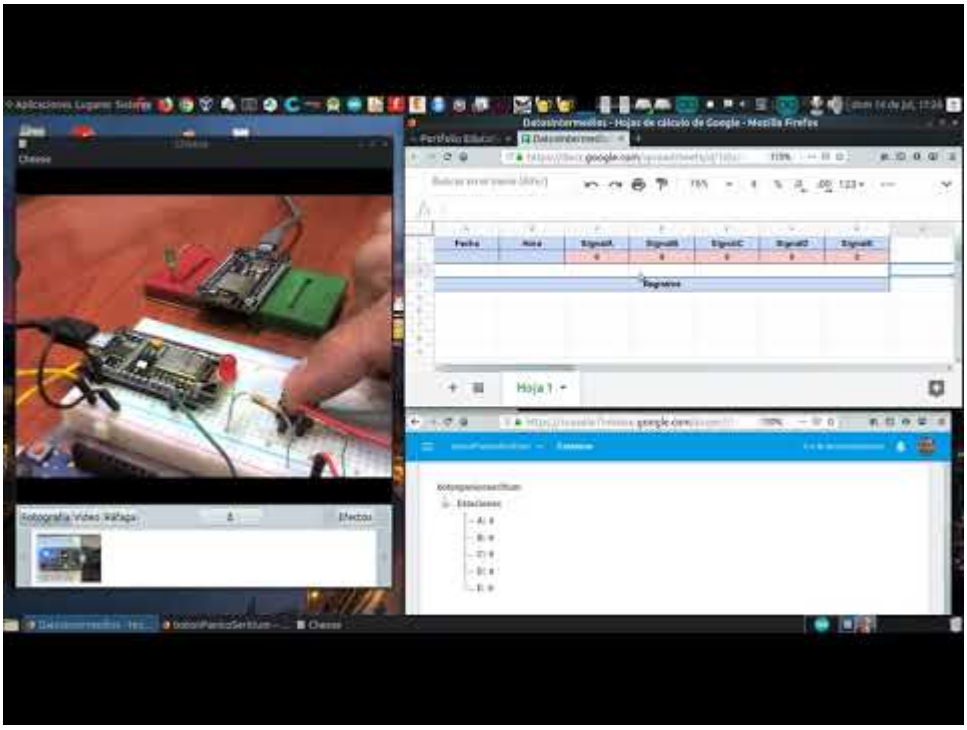
NodeMCU: comunicación con Firebase y google Spreadsheet (VII-D).

agrportfolioeducativo.blogspot.com/2019/07/nodemcu-comunicacion-con-firebase-y.html

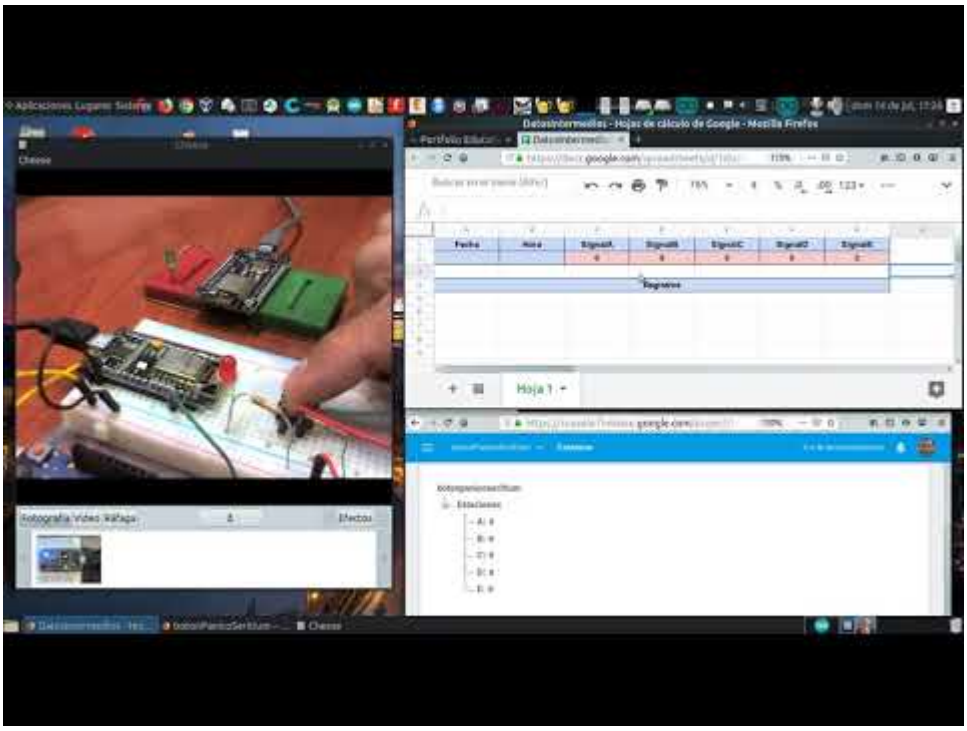


La idea principal de esta entrada es que la estación escriba el dato enviado el mismo a dos lugares: la base de datos Realtime de Firebase y la hoja de cálculo de Google Spreadsheet. La primera para una comunicación fluida con la central y con otros posibles dispositivos en un futuro; la segunda con propósitos de registro y tratamiento de datos.

Tengo dos posibilidades de hacerlo:

1. Que el NodeMCU de la estación envíe los datos a ambos lugares. Primero a Firebase y después a Google Spreadsheet.
2. Que el NodeMCU de la estación envíe los datos a Google Spreadsheet y el script de la hoja de cálculo envíe a su vez los datos a Firebase.

La primera opción me pareció más adecuada en un principio, pero también es verdad que ocuparía más espacio en la memoria del dispositivo, pues usamos 4 bibliotecas. De todas formas el código presenta muchas instrucciones con salidas al Monitor Serie que se pueden obviar.



Watch Video At: https://youtu.be/5QBMgS61Thw

Paso pues a indicar los códigos de programación....

NodeMCU

```
/*
 * Created by Aurelio Gallardo Rodríguez
 * Based on: K. Suwatchai (Mobizt)
 *
 * Email: aurelio@seritium.es
 *
 * ESTACION. Escritura. Envío de información a Firebase y Google Spreadsheet
 *
 * Julio - 2019
 *
*/

//FirebaseESP8266.h must be included before ESP8266WiFi.h
#include "FirebaseESP8266.h"
#include <ESP8266WiFi.h>
#include <WiFiClientSecure.h>
#include <ArduinoJson.h>

#define FIREBASE_HOST "[mibasededatosfirebase].firebaseio.com"
#define FIREBASE_AUTH "kfcm8934jf7894ch93c092342039h2309h" // token o secreto inventado
```

```
#define WIFI_SSID "miSSID"
#define WIFI_PASSWORD "miCONTRASEÑA"

const char* host ="script.google.com";// Este es el host de los scripts de google.
const int httpsPort =443;
// Huella digital del script de Google:
// D4:9E:40:F4:53:7A:04:93:38:F7:6B:4B:DC:70:02:A9:03:98:C2:DE
const char* fingerprint ="D4 9E 40 F4 53 7A 04 93 38 F7 6B 4B DC 70 02 A9 03 98 C2
DE";
// const char* fingerprint = "46 B2 C3 44 9C 59 09 8B 01 B6 F8 BD 4C FB 00 74 91 2F EF
F6";

String googleSheetsID ="[Identificación de Google web app]";// El que me da al
implementar una aplicación web en el script.

// objeto comunicación con Google Spreadsheet
WiFiClientSecure cliente;

//Define FirebaseESP8266 data object
FirebaseData bd;

DynamicJsonDocument doc(1024);

String path ="/Estaciones";// path a FireBase
String estacion ="A";// Estacion que voy a controlar

int i=1;// contador general

int activo=0;

#define LED 5// D1(gpio5)
#define BUTTON 4//D2(gpio4)

int estado=0;
int estadoAnterior=0;
int estadoLuz=0;

void setup()
{

  Serial.begin(115200);
  pinMode(LED, OUTPUT);
  pinMode(BUTTON, INPUT);



// conectando a la WIFI
  WiFi.begin(WIFI_SSID, WIFI_PASSWORD);
  Serial.print("Conectando a la WiFi");
```

```
while(WiFi.status()!= WL_CONNECTED)
{
   Serial.print(".");
   delay(300);
}
 Serial.println();
 Serial.print("Conectado con IP: ");
 Serial.println(WiFi.localIP());
 Serial.println();



// Conectando a la bd real Time Firebase
 Firebase.begin(FIREBASE_HOST, FIREBASE_AUTH);
 Firebase.reconnectWiFi(true);



//Set database read timeout to 1 minute (max 15 minutes)
 Firebase.setReadTimeout(bd,1000*60);
//tiny, small, medium, large and unlimited.
//Size and its write timeout e.g. tiny (1s), small (10s), medium (30s) and large (60s).
 Firebase.setwriteSizeLimit(bd,"unlimited");



 wFB(0);// Escribe un cero en la estación al arrancar



}

// Bucle principal
void loop(){

 estado=digitalRead(BUTTON);
 if (estado!=estadoAnterior){
   // el estadoAnterior ahora es el estado
   estadoAnterior=estado;
   // si estado es alto, se pasa de bajo a alto
   if (estado) {
     estadoLuz=!estadoLuz; // El estado de la luz cambia
     // sendDataToGoogleSheets(estacion,estadoLuz); // envío a googleSheet esa
información
     wFB(estadoLuz); // en vez de enviarlo a Google Spreadsheet, se lo envío a Real Time
BD Firebase
     digitalWrite(LED,estadoLuz);
     Serial.println("Estado del LED: "
```

```
String(estadoLuz));//Manda al monitor serie ese estado.
      delay(10);
      sendDataToGoogleSheets(estacion,estadoLuz);// envío a googleSheet esa información
}
}



  digitalWrite(LED,estadoLuz);
// digitalWrite(LED,rFB()); // Escribe el estado en un LED, PERO leyéndolo de la BD.
}



// Función que escribe un dato en Firebase.
// Además, actualiza el dato si tiene la misma ruta, no genera un error.
void wFB(int dato){//Write Data in FB

  if (Firebase.setInt(bd,path+"/"+estacion,dato)) {
     Serial.println("PASSED");
     Serial.println("PATH: " +bd.dataPath());
     Serial.println("TYPE: " +bd.dataType());
     Serial.println("ETag: " + bd.ETag());
     Serial.print("VALUE: ");
     if (bd.dataType() == "int")
       Serial.println(bd.intData());
     else if (bd.dataType() == "float")
       Serial.println(bd.floatData(), 5);
     else if (bd.dataType() == "double")
       printf("%.9lf\n", bd.doubleData());
     else if (bd.dataType() == "boolean")
       Serial.println(bd.boolData() == 1 ? "true" : "false");
     else if (bd.dataType() == "string")
       Serial.println(bd.stringData());
     else if (bd.dataType() == "json")
       Serial.println(bd.jsonData());
     Serial.println("----------------------------------");
     Serial.println();
  } else {
     Serial.println("FAILED");
     Serial.println("REASON: "+

bd.errorReason());
     Serial.println("----------------------------------");
     Serial.println();
}
}
```

```
// Función de lectura
int rFB(){// lee el dato de la entrada correspondiente

  if (Firebase.getInt(bd, path+"/+"

estacion)){

    if (bd.dataType() == "int") {
      Serial.println("Dato leído: "

+ (String) bd.intData());
return bd.intData();
}



}else{
    Serial.println(bd.errorReason());
}
}



/* Función de conexión. Importante la instrucción  cliente.setInsecure(); para conectar de
forma anónima
 */

void sendDataToGoogleSheets(String nombreEstacion, int Activacion){

  String cadena="";



  Serial.print("Conectando a: ");
  Serial.println(host);



// cliente.setInsecure();

    if (!cliente.connect(host, httpsPort)) {
    Serial.println("Falla la conexión a Google Sheets -->" String(host) ": "

String(httpsPort));
return;
}
```

```
if(cliente.verify(fingerprint, host)){
   Serial.println("Certificado OK");
}
else{
   Serial.println("Debes comprobar certificado");
}

      String url = "/macros/s/"+ googleSheetsID + "/exec?estacion="+ nombreEstacion+
"&activo="+

(String) Activacion;



 Serial.print("Petición  URL ");
 Serial.println(url);

 cliente.print(String("GET ") url " HTTP/1.1\r\n"
           "Host: " host "\r\n"
           "User-Agent: BuildFailureDetectorESP8266\r\n"


"Connection: close\r\n\r\n");



// Es necesario leer las cabeceras
 Serial.println("Request enviada");
while(cliente.connected()){
   String line = cliente.readStringUntil('\n');
if(line =="\r"){
    Serial.println("Cabeceras Recibidas");
    Serial.println(line);
break;
}
}

               // lee el contenido de la respuesta y lo almacena en la variable cadena
 while (cliente.available()) {
    char c = cliente.read();
    cadena = cadena

(String) c;
}
 Serial.println("Respuesta directa del servidor");
 Serial.println(cadena);
 Serial.println("==============================");
 Serial.println();
```

```
// PROBLEMA: la respuesta, por motivos de seguridad, no es jamás el texto, o el html, sino que está codificada.
// He resuelto este problema encerrando la respuesta (JSON) entre dos palabras, EMPEZAR y TERMINAR.
// Una vez localizada la información, la extraigo con la función midString.
// Los caracteres, en codificación unicode, los transformo en sus caracteres con varias líneas de código...
// Y voalá... ¡obtiene la respuesta en formato JSON de texto!
  String respuesta =midString(cadena,"EMPEZAR","TERMINAR");
  char comillas =34;
  respuesta.replace("x7b","{");
  respuesta.replace("x7d","}");
  respuesta.replace("x22",String(comillas));
  respuesta.replace("\\","");// doble slash para que lo reconozca ??
// Serial.println(respuesta);



  Serial.println("Respuesta filtrada desde el servidor");
  Serial.println(respuesta);
  Serial.println("==============================");
  Serial.println();



// Usar bibliteca JSON
  deserializeJson(doc,respuesta);
  JsonObject obj = doc.as<JsonObject>();

  String exito = obj[String("exito")];
  String comentario = obj[String("comentario")];

     Serial.println("Respuesta extraída en variables deserializando la cadena en un objeto JSON");
  Serial.println("Ha tenido éxito: "exito);
  Serial.println("Envía el comentario: "

comentario);
  Serial.println("==============================");
  Serial.println();

  delay(500);
```

```
  cliente.stop();//cierra la conexión
}

/* función texto intermedio */
String midString(String str, String start, String finish){
  int locStart = str.indexOf(start);
  if (locStart==-1) return "";
  locStart += start.length();
  int locFinish = str.indexOf(finish, locStart);
  if (locFinish==-1) return "";
  return str.substring(locStart, locFinish);
}
```

1. El código es una fusión de las llamadas a Firebase y a Google Spreadsheet que hemos visto en anteriores entradas. Lo único reseñable es que el método **cliente.setInsecure()** genera un error, pero el programa funciona sin él. No sé por qué.
2. Solo cambia la definición de **estacion="A"**; en el programa anterior estaba escrito como **estacion="/A".**

Script Web App de Google Spreadsheet.

```
function doGet(e){


var estacion = e.parameter.estacion;// variables que se han recibido. ESTACION
  estacion = estacion.toUpperCase();
var activo = e.parameter.activo;// variables que se han recibido. ACTIVO
var columna = estacion.charCodeAt(0)-62;// Código ascii -62; la "A" es 65, luego da el valor 3, correspondiente a la tercera columna


// Dirección de la base de datos RealTime de Firebase
var bd ="[mibasededatosfirebase].firebaseio.com";
// Secreto de la base de datos Firebase
var secreto ='lkfjweo45roi56oifUIUEoierfioerfncoc'; // inventado
// Ruta de la base de datos RealTime
var ruta ="Estaciones";

        var options = { method:'get', contentType: 'application/json' }; //opciones de la llamada a fetch
  var firebaseURL = 'https://'+bd+'/'+ruta+'.json?auth='+

secreto;// construyo la ruta
```

```
if(columna>=3&& columna<=7&& estacion.length==1){// si es una sola letra, una
columna
var resultado = UrlFetchApp.fetch(firebaseURL, options);



// Hoja de cálculo activa
var sheet = SpreadsheetApp.getActiveSheet();
var ultimalinea = sheet.getLastRow();// obtiene la última línea



// Fechas y horas actuales.
var fecha=Utilities.formatDate(new Date,"GMT+1","dd/MM/yyyy");
var d =new Date();
var hora = d.toLocaleTimeString();



// Objeto JSON según el resultado



var e = JSON.parse(resultado);

      for (clave in e) {
    var columna = clave.charCodeAt(0)-62; // columna de cada valor
    var valorActual = sheet.getRange(2,columna).getValue();
    if (valorActual!=e[clave]) { // Si los valores son distintos
      // 1) Cambio fecha y hora en la fila dos
      sheet.getRange(2,1).setValue(fecha);
      sheet.getRange(2,2).setValue(hora);
      // 2) Cambio valor actual en la fila dos
      sheet.getRange(2,columna).setValue(parseInt(e[clave]));
      // 3) Pongo fecha y hora en la última línea
      sheet.getRange(ultimalinea1,1).setValue(fecha);
      sheet.getRange(ultimalinea1,2).setValue(hora);
      // 4) Escribo en la última línea un registro de lo que acontece
      segunActivo(clave,e[clave],ultimalinea

1,columna,sheet);
}
```

```
// Logging de los datos...
     Logger.log(clave);
     Logger.log(e[clave]);



}// Fin del for

       // *** Refresco ***
   SpreadsheetApp.flush();

var output =
HtmlService.createHtmlOutput('EMPEZAR'JSON.stringify(construyeJSON("1","Estación
"estacion" activo: "activo))'TERMINAR');

   // output.setXFrameOptionsMode(HtmlService.XFrameOptionsMode.ALLOWALL);



return output;



}else{// Fin del if que reconoce la columna



   Logger.log("Estación fuera del rango");



}



}

// funcion que escribe en el registro según si está activo o no
function segunActivo(estacion,dato,fila,columna,sheet) {
  if (dato==1) {
     sheet.getRange(fila,columna).setValue("Estación "estacion" se activa");
     sheet.getRange(fila,columna).setFontColor("darkgreen");
     sheet.getRange(fila,columna).setBackground("#AAFFAA");
  } else if (dato==0) {
     sheet.getRange(fila,columna).setValue("Estación "estacion

" se desactiva");
     sheet.getRange(fila,columna).setFontColor("darkred");
     sheet.getRange(fila,columna).setBackground("#FFAAAA");
```

```
}
}

  function construyeJSON(success, comentario) {
  var json = {
    'exito':success,
    'comentario':comentario,
  };
  return json;
}
```

1. El código del script es también una fusión de scripts a los que ya nos hemos referido.
2. Básicamente recibe una petición GET (por eso el script es una función doGet) y extraigo los valores de la estación, y si está activo o no.
3. Si tengo un valor correcto de estación, accedo a Firebase y leo los datos registrados (que deberían haberse escrito ANTES desde el NodeMCU), detecta los cambios, y, si estos existen los refleja en la hoja de cálculo.

= = = = = = = = =