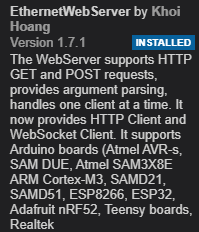
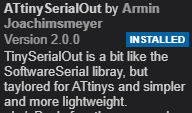
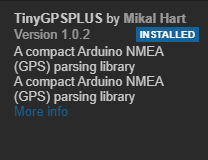
Librerías instaladas



Definición de librerías

//---------Librerías------------

#include <TinyGPS++.h> // Librería para el módulo GPS.

#include <SoftwareSerial.h> //librerías necesarias para la conexión WiFi

#include <ESP8266WiFi.h> //       ""

#include <ESP8266HTTPClient.h>//       ""

Definición de Variables

//---------Variables------------

#define PIN\_ANALOG\_RAIN\_SENSOR A0  //Entrada analógica para la señal del sensor lluvia

TinyGPSPlus gps;  //Objeto de la librería TinyGPS++

SoftwareSerial ss(4, 5); //Conexión serial del GPS a los pines GPIO4 y 5 de la placa NodeMCU

const char\* ssid = "Telecentro-Galarza";//Nombre de la red WiFi

const char\* password = "0111171227";//Contraseña de la red WiFi

String serverName = "http://localhost:4000"; //Nombre del Servidor de la página

float latitude , longitude; //Definición de las variables dónde se almacenarán la longitud y latitud

int year , month , date, hour , minute , second; //Variables donde se almacenarán los datos del Año, mes, día, hora, minuto y segundo

String date\_str , time\_str , lat\_str , lng\_str; //Definición de las variables dónde se almacenarán la fecha, la hora, la latitud y la longitud

int pm; //Definición de la variable PM

WiFiServer server(80);  //Servidor WiFi

Función Setup

void setup()

{

  Serial.begin(115200);//inicio de la comunicación serie

  ss.begin(9600);//inicio de la comunicación con el GPS

  Serial.println();

  Serial.print("Connecting to ");

  Serial.println(ssid);

  WiFi.begin(ssid, password); //Conectando al WiFi

  while (WiFi.status() != WL\_CONNECTED)// Mientras no está conectado a una red

  {

    delay(500);

    Serial.print("."); //Imprime "...."

  }

  Serial.println("");

  Serial.println("WiFi connected");

  server.begin();

  Serial.println("Server started");

  Serial.println(WiFi.localIP());  // Imprimir la IP

}

Función Loop

Latitud y longitud

void loop(){

while (ss.available() > 0) //Cuando los datos están disponibles

    if (gps.encode(ss.read())) //Leer datos del GPS

    {

      if (gps.location.isValid()) //Si los datos de la localización que el gps lee son correctos

      {

        latitude = gps.location.lat();

        lat\_str = String(latitude , 6); // Valor de la latitud en una cadena de caracteres

        longitude = gps.location.lng();

        lng\_str = String(longitude , 6);  // Valor de la longitud en una cadena de caracteres

      }

Fecha

if (gps.date.isValid()) //Si los datos (día, mes y año) que el gps lee son válidos

      {

        date\_str = "";

        date = gps.date.day();

        month = gps.date.month();

        year = gps.date.year();

        if (date < 10)

          date\_str = '0';

        date\_str += String(date); // Valor del día guardado en una cadena de caracteres

        date\_str += " - ";

        if (month < 10)

          date\_str += '0';

        date\_str += String(month); // Valor del mes guardado en una cadena de caracteres

        date\_str += " - ";

        if (year < 10)

          date\_str += '0';

        date\_str += String(year); // Valor del Año guardado en una cadena de caracteres

      }

Tiempo

if (gps.time.isValid())  //Si el tiempo que el gps está leyendo es válido

      {

        time\_str = "";

        hour = gps.time.hour();

        minute = gps.time.minute();

        second = gps.time.second();

        if (minute > 59) // reseteo de minutos si los minutos pasan de 59 (60) se resetea, entonces nos muestra el minuto "0"

        {

          minute = minute - 60;

          hour = hour + 1;

        }

        hour = (hour - 3) ;  //Argentina tiene un formato UTC o GMT de -3 horas

        if (hour > 23)

          hour = hour - 24;   // reseteo de horario si la hora es más de 23 (24) se resetea, entonces nos muestra la hora "0"

        if (hour >= 12)  // Saber si es PM o AM

          pm = 1;

        else

          pm = 0;

        hour = hour % 12;

        if (hour < 10)

          time\_str = '0';

        time\_str += String(hour); //Valor de horas guardados en una cadana de caracteres.

        time\_str += " : ";

        if (minute < 10)

          time\_str += '0';

        time\_str += String(minute);//Valor de minutos guardados en una cadana de caracteres.

        time\_str += " : ";

        if (second < 10)

          time\_str += '0';

        time\_str += String(second); //Valor de segundos guardados en una cadana de caracteres

        if (pm == 1)

          time\_str += " PM ";

        else

          time\_str += " AM ";

      }

    }

Conexión HTTP

WiFiClient client = server.available(); // Pedimos el estado del cliente

  if (!client)  //Preguntamos si está desconectado

  {

    return; //Enviamos nada

  }

  HTTPClient http; //Pedimos una conexión HTTP

  int sensorValue = analogRead(PIN\_ANALOG\_RAIN\_SENSOR); // Leer datos del puerto analógico (sensor de agua)

  //Creamos una ruta (http://localhost:4000/GPS?fecha=\*Una fecha\*&tiempo=\*un horario\*&latitud=\*una latitud\*&longitud=\*una longitud\*&agua=\*detección de agua\*)

  String serverPath = serverName + "/GPS?fecha=" + date\_str + "&tiempo=" + time\_str + "&latitud=" + lat\_str + "&longitud=" + lng\_str + "&agua=" + sensorValue;

  http.begin(client, serverPath.c\_str()); //iniciamos la conexión HTTP

  int httpResponseCode = http.GET(); //Creamos una variable que almacene una petición de tipo GET

if (httpResponseCode>0) { //si la conexión existe continúa con el proceso

    Serial.print("HTTP Response code: ");

    Serial.println(httpResponseCode);

    String payload = http.getString();

    Serial.println(payload);

  }

  else { //si la conexión no existe retorna un error

    Serial.print("Error code: ");

    Serial.println(httpResponseCode);

  }

  http.end(); //terminamos la conexión HTTP

  delay(5000); // Retardo entre mediciones

}