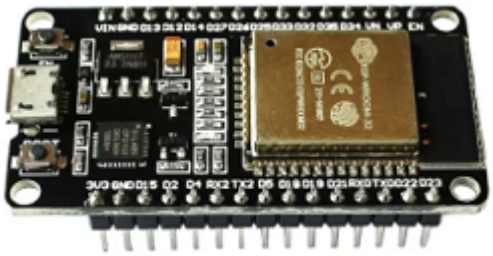
(https://www.prometec.net/)

Objetivos

- Vamos a configurar el brokerMosquitto en Raspberry Pi.
- Añadiremos unas pocas líneas al fichero de configuración.
- Veremos como crear usuario y contraseñas.
- Aprovecharemos para presentar el MQTT explorer.*[/three-fourth] [clear/]*

Material requerido.

	**Un ESP32 (https://store.prometec.net/producto/esp32-wifi-bluetooth/)**
MQTT	Acceso a un broker MQTT

Publicando con ESP32 en un broker MQTT

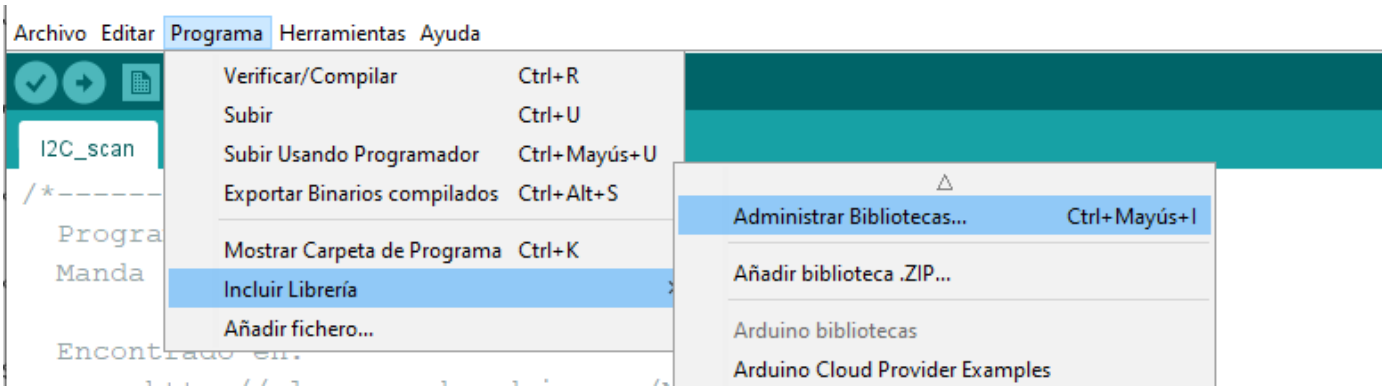
En las sesiones previas hemos montado y configurado un **servidor MQTT** con **Mosquitto** para poder publicar a su través datos desde nuestros servidores. Vimos como publicar y suscribir topics con línea de comandos y un poco por encima como usar el **MQTT Explorer** para suscribir o publicar de forma gráfica, pero lo que no habíamos visto hasta ahora, era como conectar sus Arduinos a un Broker MQTT

No es que sea una gran novedad este programa. Internet esta lleno de ejemplos (*Casi todos el mismo*) pero haremos una muy pequeña variación publicando valores aleatorios en un topic propio, para ilustrar como publicar lecturas numéricas (*Como medidas de un sensor*) en lugar de textos que ilustran el método básico, y un poco más adelante haremos otro ejemplo programando un sensor como el SCD41 de temperatura, humedad y CO_2 como ejemplo con valores reales.

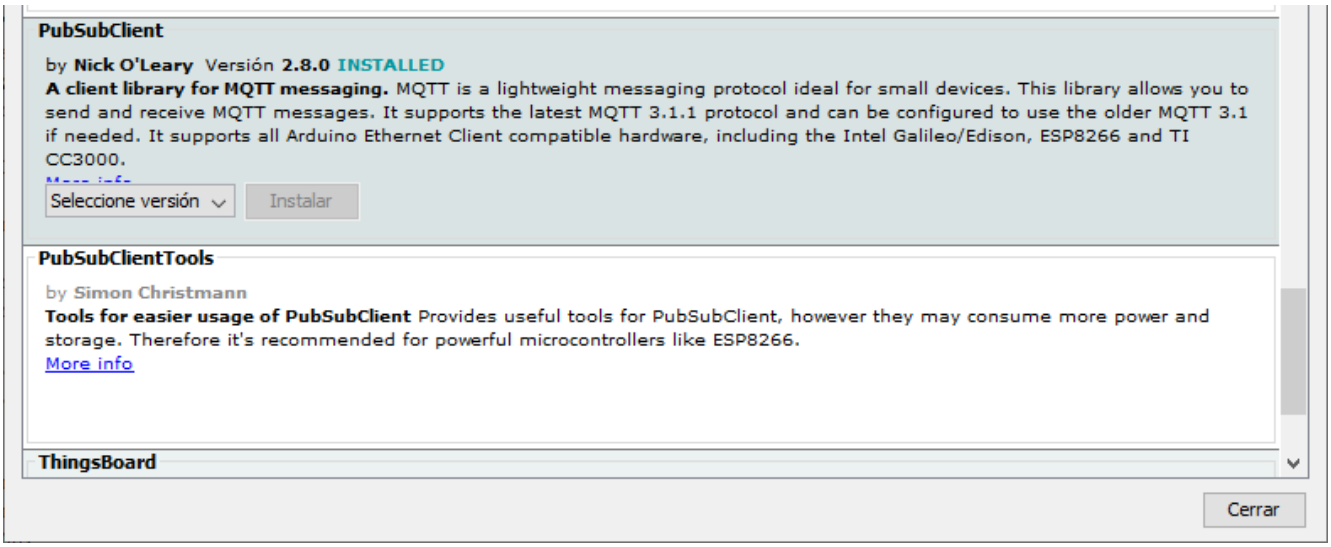
Pero de momento vamos a empezar con la versión más simple posible publicando valores numéricos aleatorios en nuestro recién instalado broker MQTT para empezar a jugar con la librería.

Instalar las librerías MQTT

Como viene siendo habitual necesitaremos un par de librerías para conectar nuestro ESP32 a un **broker MQTT**. Por un lado, la de **WIFI** y por otro el cliente de **MQTT**, para lo que usaremos la librería PubSubClient que podemos instalar directamente desde el administrador de bibliotecas:



Una vez ahí, escribe PubSubClient en la ventana de búsqueda y busca la versión de Nick O'Leary y dale a instalar (*La mía ya está instalada, porque ya tengo hechos los deberes*)



Con esto ya podemos empezar con nuestro programa

Conectando ESP32 alBroker MQTT

Vamos a empezar con los includes

```
#include <WiFi.h>
#include <PubSubClient.h>
```

Para conectar a un servidor MQTT desde ESP32 vamos a usar WIFI por lo que vamos a necesitar unas cuantas cosas como SSID y pass de la WIFI, además de un usuario registrado con contraseña en el broker y su dirección (IP o URL) el puerto… etc.

```
const char* ssid = "REDLINEB";
const char* password = "contrase";
const char* mqttServer = "192.168.1.36";
const int mqttPort = 1883;
const char* mqttUser = "charly";
const char* mqttPassword = "contrase";
```

```
WiFiClient espClient;
PubSubClient client(espClient);
```

Ya podemos empezar con el setup() que es donde efectuaremos la conexión:

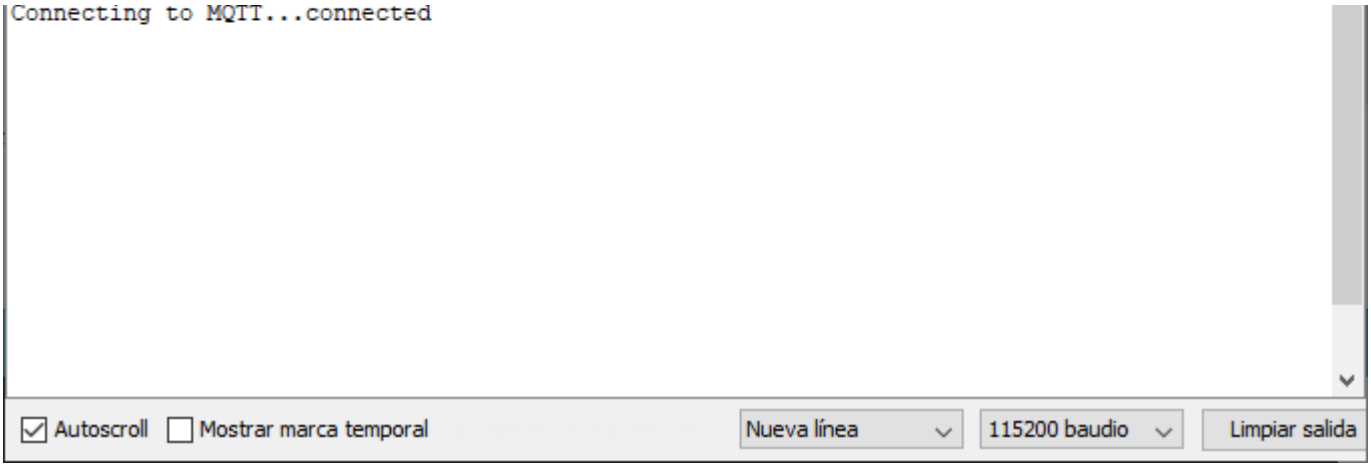
```
void setup()
{ Serial.begin(115200);
  WiFi.begin(ssid, password);
  Serial.println("................................");

  Serial.print("Connecting to WiFi.");
  while (WiFi.status() != WL_CONNECTED)
       {  delay(500);
          Serial.print(".") ;
       }
  Serial.println("Connected to the WiFi network");
```

Nada nuevo hasta aquí, una conexión WIFI habitual con el ESP32. Ahora toca conectar al servidor MQTT *(Y esto si es nuevo)*:

```
client.setServer(mqttServer, mqttPort);
while (!client.connected())
{      Serial.println("Connecting to MQTT...");
       if (client.connect("ESP32Client", mqttUser, mqttPassword ))
           Serial.println("connected");
       else
       {   Serial.print("failed with state ");
           Serial.print(client.state());
           delay(2000);
       }
}
```

Si todo va bien verás algo así:



Una vez que hayamos conectado ya podemos publicar en el broker MQTT.... Con el loop:

```
void loop()
  {  client.loop();
     char str[16];
     sprintf(str, "%u", random(100));

     client.publish("Prosensor/CO2", str);
     Serial.println(str);
     delay(500);
 }
```

La primera instrucción es importante: client.loop(), comprueba que no haya mensajes pendientes de entrada y mantiene la conexión con el server. No puedes olvidarla, mientras que la instrucción para publicar un topic es:

```
client.publish( topic, mensaje);
```

Pero como queremos publicar una lectura numérica podríamos caer en el error de intentar publicar así, un número aleatorio:

```
client.publish("Prosensor/CO2", String(random(100)));
```

Algo que provocaría un ladrido instantáneo del compilador:

```
no matching function for call to 'PubSubClient::publish(const char [14], String)'
```

(https://www.prometec.net/)

```
Se encontraron varias bibliotecas para "WiFi.h"
Usado: C:\Users\usuario\AppData\Local\Arduino15\packages\esp32\hardware\esp32\1.0.6\libraries\WiFi
 No usado: C:\Program Files (x86)\Arduino\libraries\WiFi
exit status 1
no matching function for call to 'PubSubClient::publish(const char [14], String)'
```

48 ESP32 Dev Module, Disabled, Default 4MB with spiffs (1.2MB APP/1.5MB SPIFFS), 240MHz (WiFi/BT), QIO, 80MHz, 4MB (32Mb), 921600,

La razón es que la librería no permite publicar Strings *(Que sería cómodo)*, sino que tienen que ser constant char[] y por eso hemos empezado definiendo:

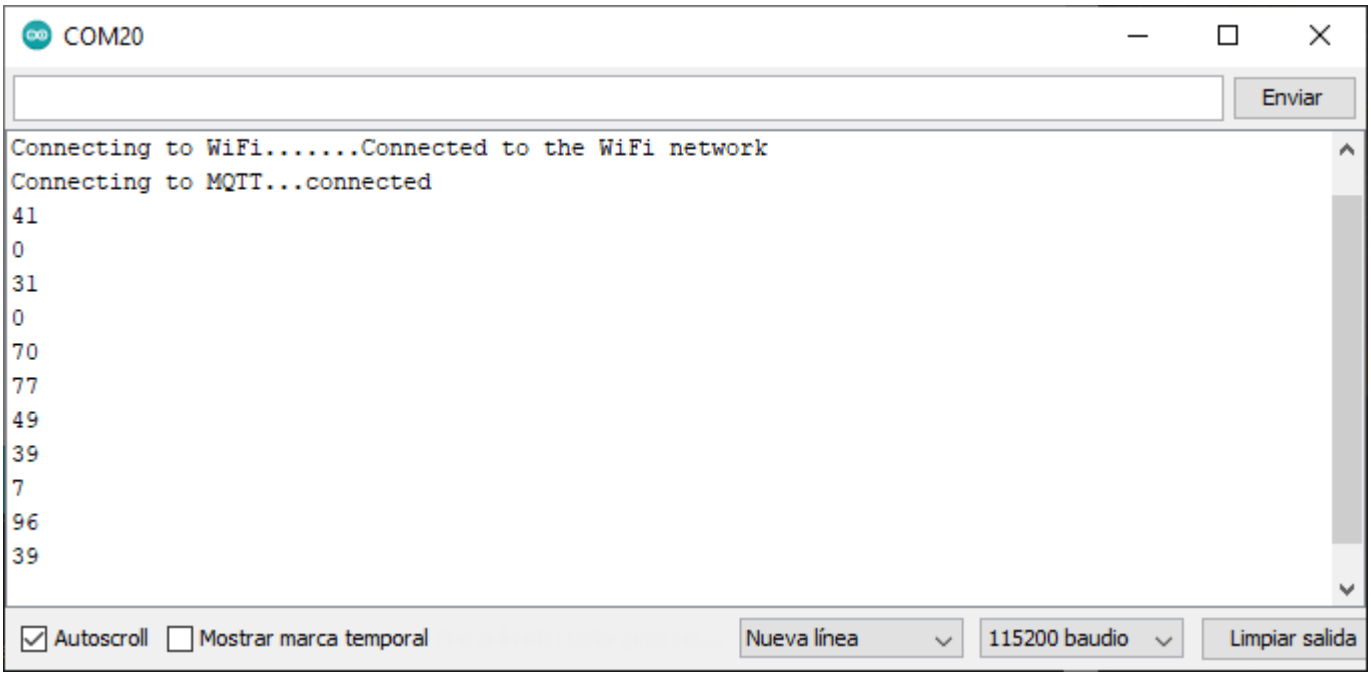
```
char str[16];
sprintf(str, "%u", random(100));
```

Donde sprintf() es una función que asigna el valor de lo que queramos, un entero random, a un char array con el formato que nos interesa.*(La u del formato es de unsigned y todos contentos)*

Aquí os dejo el programa completo:

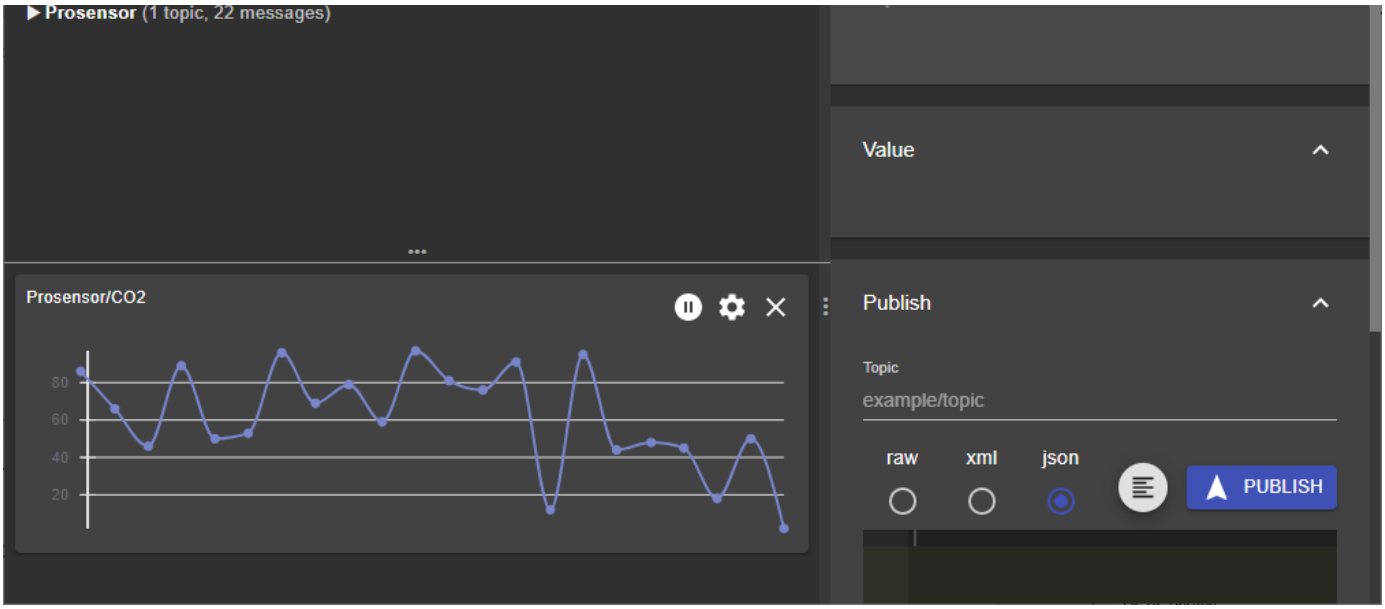
MQTT_test (https://www.prometec.net/wp-content/uploads/2021/12/MQTT_test.rar)

Ahora nuestro **ESP32** empezará a publicar los mensajes simulados en el Broker MQTT con copia al consola para poder comprobar que la cosa marcha:



De este modo podemos ver en la consola serie los valores que vamos publicando y a la vez usar el MQTT Explorer para suscribir esos mismos valores (*Que es lo que hicimos en el la ultima sesión para ver el uso del mqtt explorer)*:

(https://www.prometec.net/)



Copyright © 2023 - Prometec - All rights reserved. Powered by WordPress

(https://www.prometec.net)

(https://www.prometec.net/)