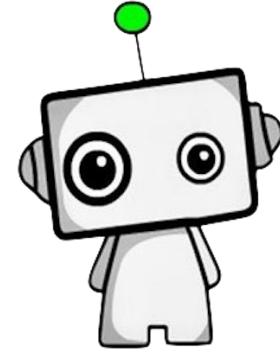


# Smartknob



# Index

- 0. Resumen
- 1. Prefacio
- 2. Objetivo
- 5. Hardware
- 6. Software
- 7. Discussion



# Resumen

**knob** *sustantivo*

**perilla** *f* ⓘ (plural: *perillas* *f*)

I turned the volume knob of the radio set.

Giré la perilla del volumen de la radio.

**botón** *m* ⓘ (plural: *botones* *m*)

**pomo** *m* ⓘ (plural: *pomos* *m*)

I turned the knob and opened the door.

Giré el pomo y abrí la puerta.

*menos frecuente:*

**manilla** *f* ⓘ · **tirador** *m* ⓘ · **puño** *m* ⓘ · **picaporte** *m*

*Ejemplos:*

rotary knobs *pl* — botones giratorios *pl m* ⓘ

control knob *s* — botón regulador *m* · perilla de control *f*

locking knob *s* — perilla de bloqueo *f* ⓘ · perilla de fijación *f*

# Prefacio

MIDI (siglas de Musical Instrument Digital Interface o en español “Interfaz Digital de Instrumentos Musicales”) es un estándar tecnológico que describe un protocolo, una interfaz digital y conectores que permiten que varios instrumentos musicales electrónicos, ordenadores y otros dispositivos relacionados se conecten y comuniquen entre sí.

Una simple conexión MIDI puede transmitir hasta dieciséis canales de información que pueden ser conectados a diferentes dispositivos cada uno.

Existen una gran cantidad de dispositivos analógicos que no tienen implementación MIDI, por lo que no entienden este tipo de señal, y que son controlados a través de potenciómetros manualmente o por control de voltaje (CV).

# Objetivo

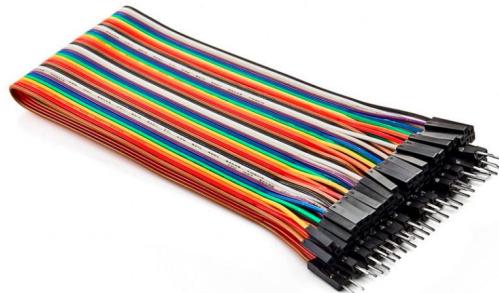
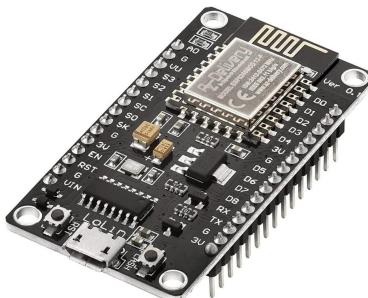
La idea es automatizar la rotación de una perilla con ayuda de un motor a través de mensajes MIDI transmitidos por WiFi.



Figura 1: diagrama del proyecto

# Hardware

- ESP8266
- Servo sg90
- Cables dupont
- Base a medida para imantado



# Modelo v1

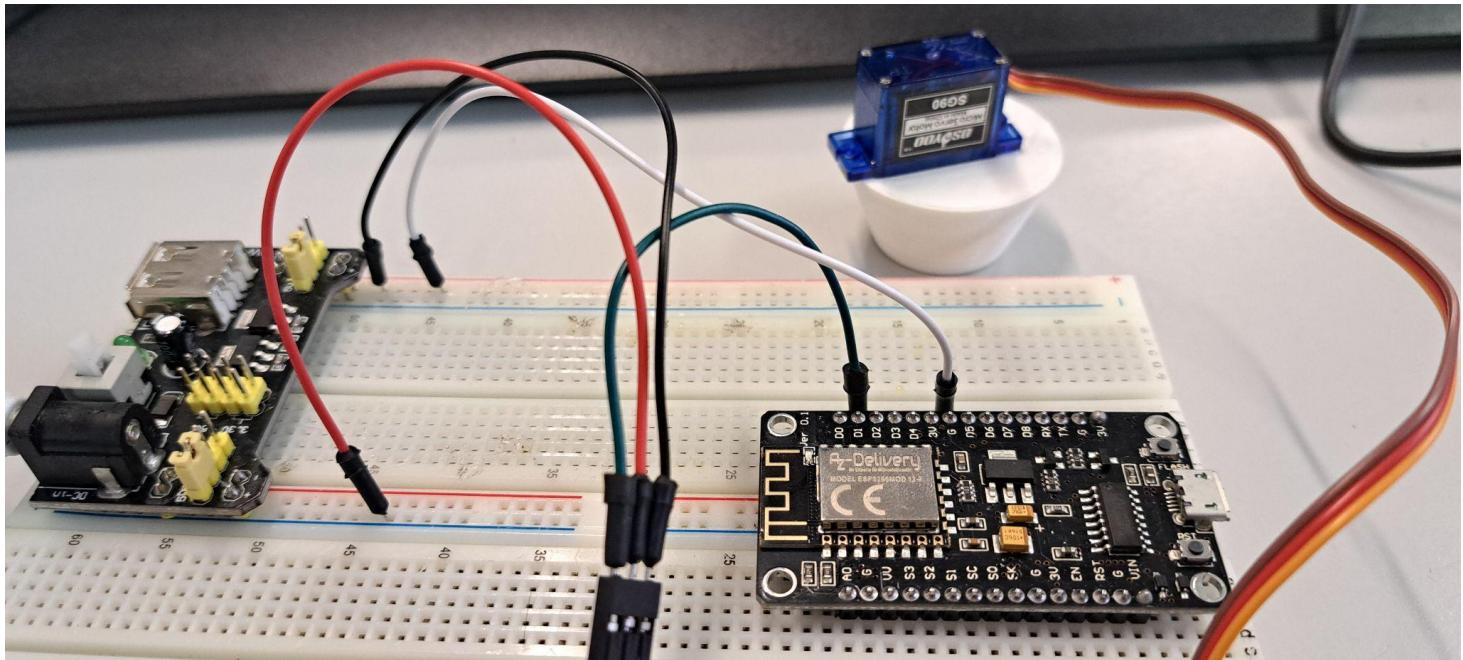


Figura 2: Prototipo 1. ESP8266 + servo SG90 + hv131

# Modelo v1.1

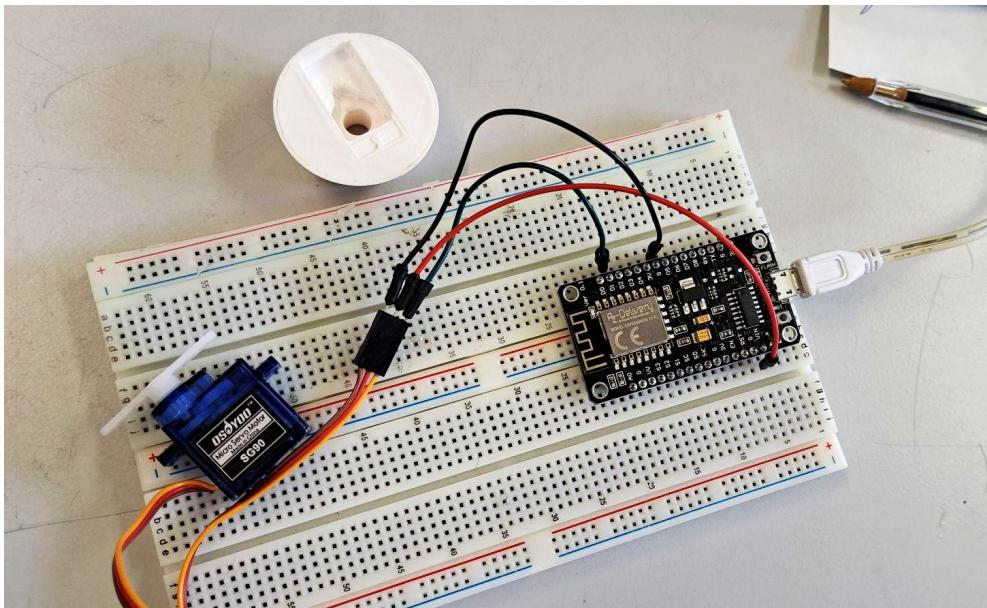


Figura 3: Prototipo 2. La ESP8266 entrega 5V por el pin Vin, así que quitamos el hv131

# Modelo v2

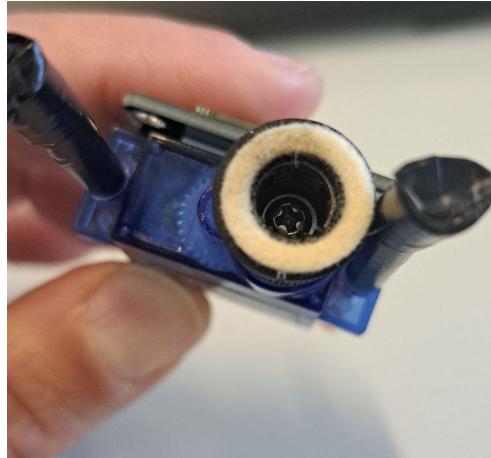
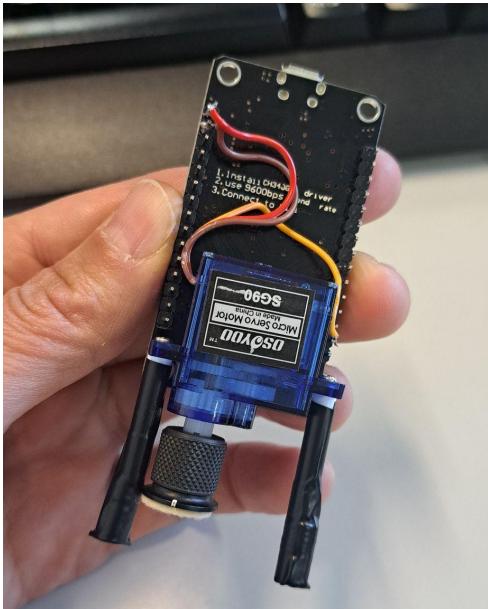
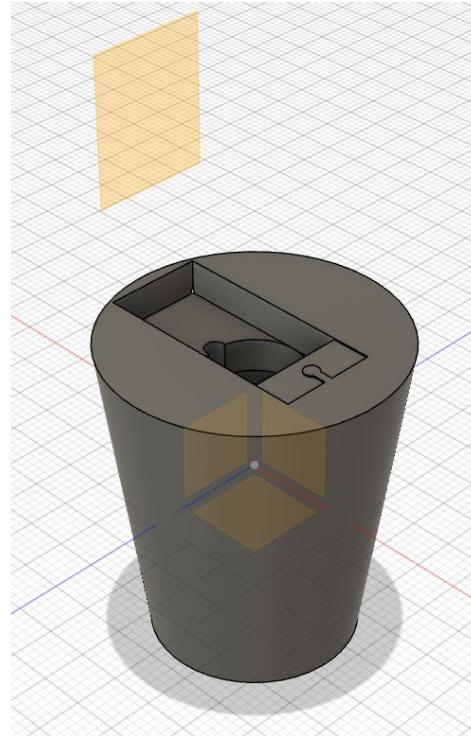
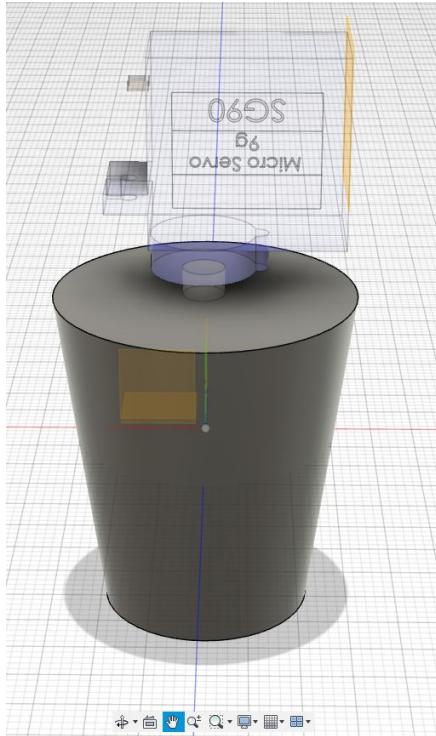
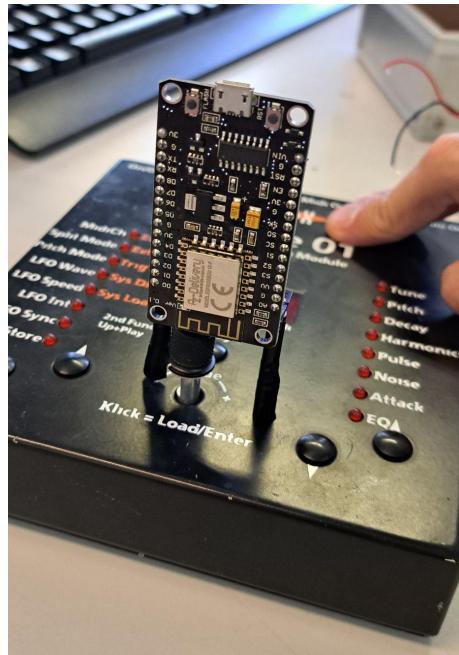


Figura 3: Prototipo 3. Se incrusta el servo en la ESP8266 a un par de pilares imantados.

# Modelo 3D



# Modelo v2



# Software

- Cubase emite señal MIDI.
- rtpMIDI envía por WiFi la señal MIDI
- Autodesk fusion 360 para generar modelos 3D de las piezas necesarias.
- Arduino IDE para escribir, compilar y transferir el código a la microcontroladora.
  - a. Añadir json en preferencias (figura 7)  
[http://arduino.esp8266.com/stable/package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json)
  - b. Añadir la librería esp8266 by ESP8266 Community
  - c. Seleccionar la board (figura 8)

# Cubase

- Cubase emite señal MIDI: hay que crear un canal MIDI y asignar a su salida el puerto virtual creado en rtpMIDI. A continuación se asigna a esta pista MIDI el canal 1 u omni.

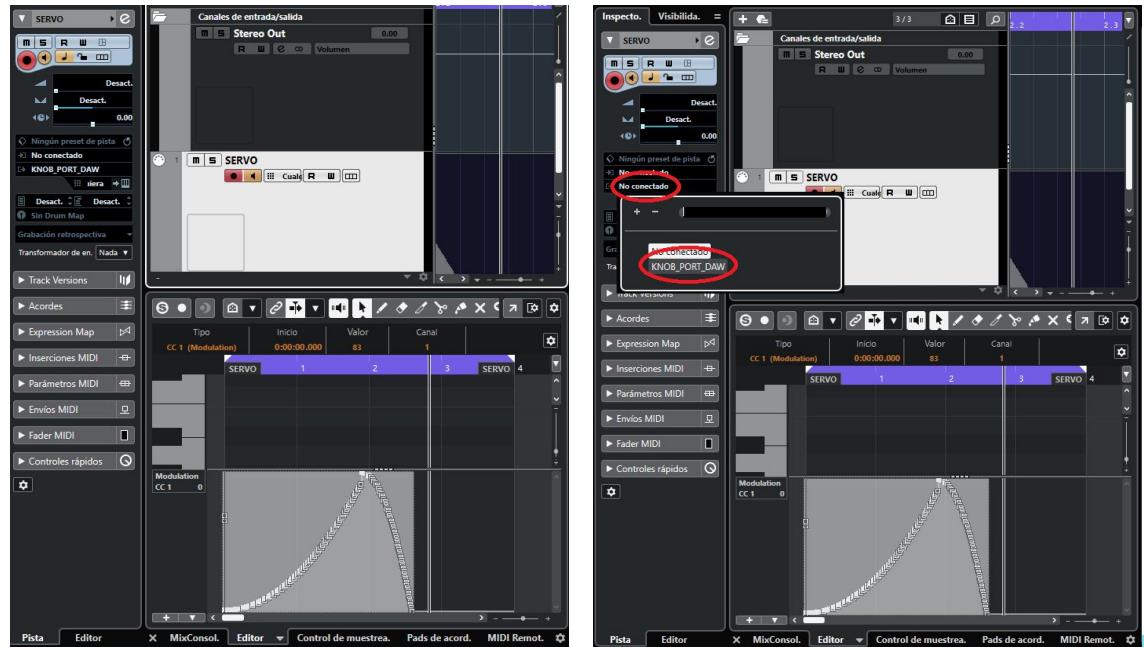


Figura 4: Cubase

# rtpMIDI I

- rtpMIDI envía por WiFi la señal MIDI. Hay que crear una “nueva sesión”, de esta forma se crea un nuevo puerto MIDI virtual visible y assignable desde el DAW (Digital Audio Workstation)

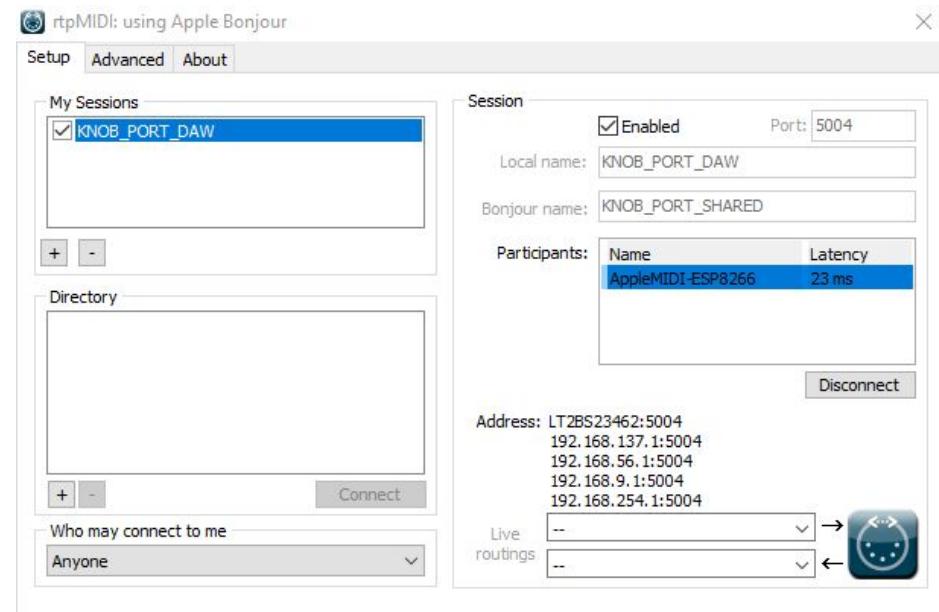


Figura 5: Creación de sesión midi desde rtpMIDI

# rptMIDI II

- Se crea una nueva instancia en “Directory” y se le asigna la IP de la ESP8266 y el puerto que indica la “Session”

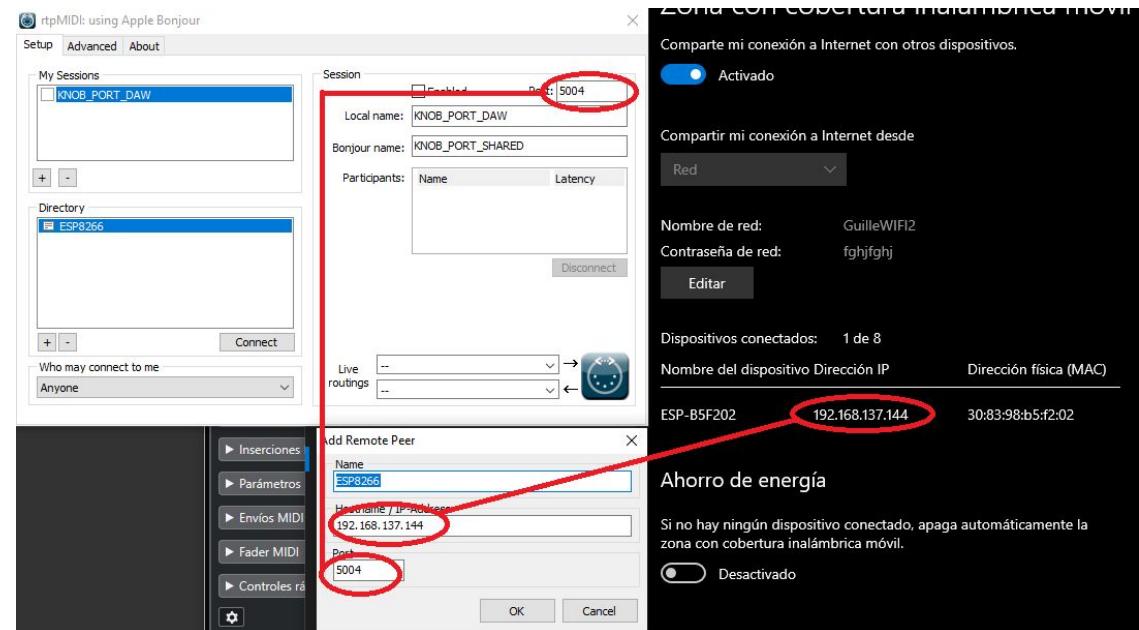


Figura 6: Configuración de instancias de rptMIDI

ServoKnob | Arduino IDE 2.2.1

File Edit Sketch Tools Help

Select Board

BOARDS MANAGER

esp8266 by ESP8266 Community

Boards included in this package: Generic ESP8266 Module, Generic ESP8285 Module, Lilely Agrumino Lemon v4, ESPDuino (ESP-13 Module), Adafruit...  
More info

3.1.2 **INSTALL**

ServoKnob.ino

```

1 #include <ESP8266WiFi.h>
2 #include <WiFiClient.h>
3 #include <WiFiUdp.h>
4 #include <Servo.h>
5 #define SerialMon Serial
6 #define MIDI2SERVO 1.414
7 #include <AppleMIDI_Debug.h>

```

Preferences

Sketchbook location: c:\Users\AdministradorCIFO\Documents\Arduino **BROWSE**

Show files inside Sketches

Editor font size: 14  Automatic 100 %

Interface scale:  Light

Theme: English (Reload required)

Language: English (Reload required)

Show verbose output during  compile  upload

Compiler warnings: None

Verify code after upload  
 Auto save  
 Editor Quick Suggestions

Additional boards manager URLs: index.json, http://arduino.esp8266.com/stable/package\_esp8266com\_index.json **+**

CANCEL OK

DBG("Connected to the network");
DBG(F("OK, now make sure you have an rtpMIDI session that is Enabled"));
DBG(F("Add a device named Arduino with Host"), WiFi.localIP(), "Port", AppleMIDI.getPort(), "(Name", Apple
DBG(F("Select and then press the Connect button"));
DBG(F("Then open a MIDI sender and send MIDI CC1 (Control Change 1) messages to this device"));

MIDI.begin();

AppleMIDI.setHandleConnected([](const APPLEMIDI\_NAMESPACE::ssrc\_t & ssrc, const char\* name) {

Figura 7: IDE Arduino para ESP8266. Paso 1.

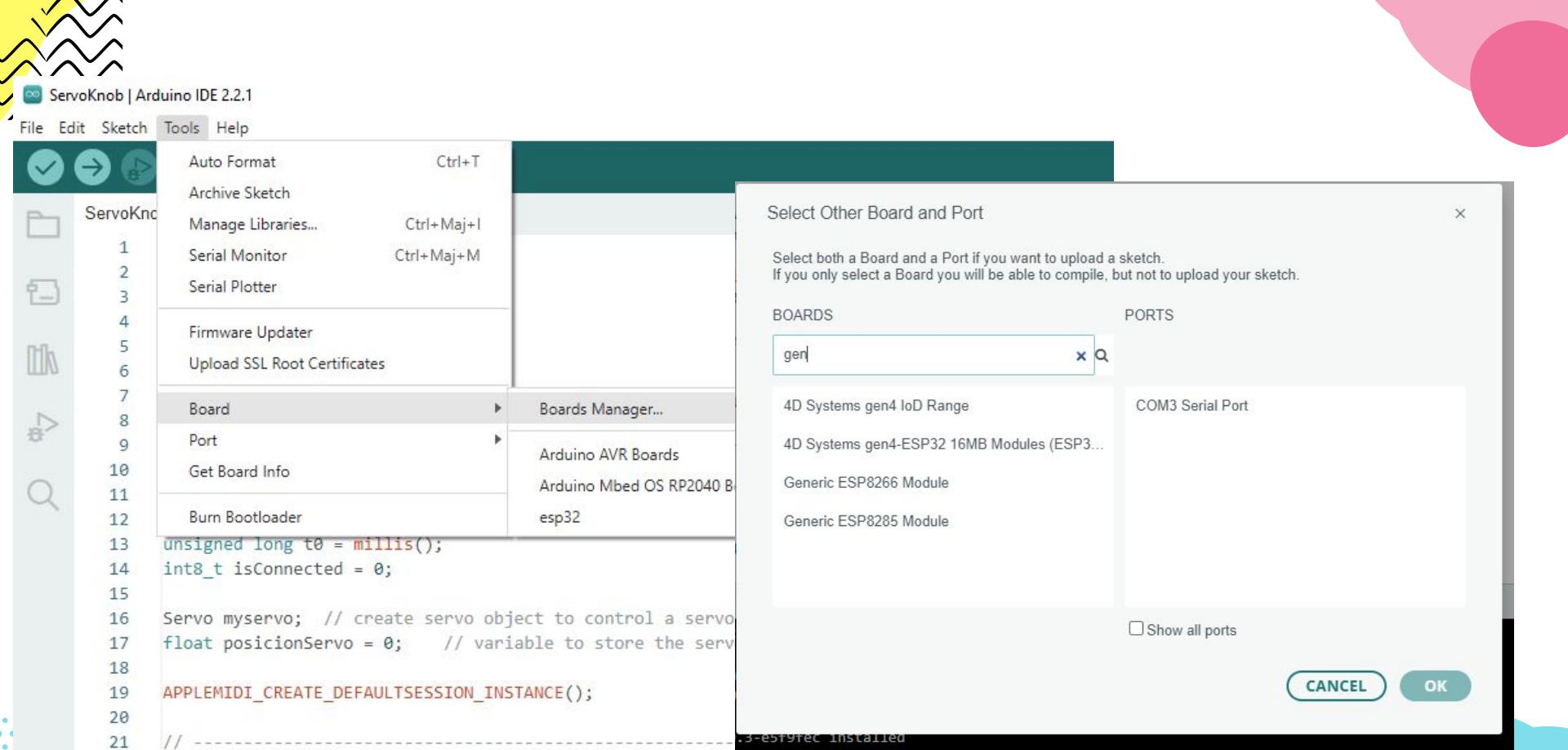


Figura 8: IDE Arduino para la ESP8266. Paso 2.



Generic ESP8266 Mod... ▾

```
ServoKnob.ino

1 #include <ESP8266WiFi.h>
2 #include <WiFiClient.h>
3 #include <WiFiUpd.h>
4 #include <Servo.h>
5 #define SerialMon Serial
6 #define MIDI2SERVO 1.414
7 #include <AppleMIDI_Debug.h>
8 #include <AppleMIDI.h>
9
10 char ssid[] = "GuilleWIFI2"; // your network SSID (name)
11 char pass[] = "fghjfghj"; // your network password (use for WPA, or use as key for WEP)
12
13 unsigned long t0 = millis();
14 int8_t isConnected = 0;
15
16 Servo myservo; // create servo object to control a servo
17 float posicionServo = 0; // variable to store the servo position
18
19 APPLEMIDI_CREATE_DEFAULTSESSION_INSTANCE();
20
21 // -----
22 //
23 // -----
24 void setup()
25 {
26     myservo.attach(5, 544, 2400); // attaches the servo on pin 5 to the servo object
27     DBG_SETUP(115200);
28     DBG("Booting");
29
30     WiFi.begin(ssid, pass);
```

```
31
32     while (WiFi.status() != WL_CONNECTED) {
33         delay(500);
34         DBG("Establishing connection to WiFi..");
35     }
36     DBG("Connected to the network");
37     DBG(F("OK, now make sure you have an rtpMIDI session that is Enabled"));
38     DBG(F("Add a device named Arduino with Host"), WiFi.localIP(), "Port", AppleMIDI.getPort(), "(Name", AppleMIDI.getName(), ")");
39     DBG(F("Select and then press the Connect button"));
40     DBG(F("Then open a MIDI sender and send MIDI CC1 (Control Change 1) messages to this device"));
41
42     MIDI.begin();
43
44     AppleMIDI.setHandleConnected([](const APPLEMIDI_NAMESPACE::ssrc_t & ssrc, const char* name) {
45         isConnected++;
46         DBG(F("Connected to session"), ssrc, name);
47     });
48     AppleMIDI.setHandleDisconnected([](const APPLEMIDI_NAMESPACE::ssrc_t & ssrc) {
49         isConnected--;
50         DBG(F("Disconnected"), ssrc);
51     });
52
53     // Manejar mensajes MIDI CC1 (Control Change 1)
54     MIDI.setHandleControlChange([](byte channel, byte controller, byte value) {
55         if (controller == 1) {
56             DBG(F("Received MIDI CC1 (Control Change 1) with value"), value);
57             posicionServo = value * MIDI2SERVO; // Incrementa la posición del servo en 1.414 grados por paso de CC1
58             DBG(F("TENEMOS EL VALOR "), posicionServo);
59             myservo.write(posicionServo);           // tell servo to go to position in variable 'pos'
60             delay(30);                          // waits 15 ms for the servo to reach the position
61         }
62     });
63 }
64
65 void loop() {
66     // Escuchar mensajes MIDI entrantes
67     MIDI.read();
68     // No se envía ningún mensaje MIDI en este código
69     // Resto del código...
70 }
```