

# Librerías instaladas

## esp8266

by **ESP8266 Community** versión **3.0.2** **INSTALLED**

Tarjetas incluidas en este paquete

Generic ESP8266 Module, Generic ESP8285 Module, Lively Agrumino Lemon v4, ESPDuino (ESP-13 Module), Adafruit Feather HUZZAH ESP8266, Invent One, XinaBox CW01, ESPresso Lite 1.0, ESPresso Lite 2.0, Phoenix 1.0, Phoenix 2.0, NodeMCU 0.9 (ESP-12 Module), NodeMCU 1.0 (ESP-12E Module), Olimex MOD-WIFI-ESP8266(-DEV), SparkFun ESP8266 Thing, SparkFun ESP8266 Thing Dev, SparkFun Blynk Board, SweetPea ESP-210, LOLIN(WEMOS) D1 R2 & mini, LOLIN(WEMOS) D1 mini (done), LOLIN(WEMOS) D1 mini Pro, LOLIN(WEMOS) D1 mini Lite, LOLIN(WeMos) D1 R1, ESPino (ESP-12 Module), ThaiEasyElec's ESPino, WifiInfo, Arduino, 4D Systems gen4 IoT Range, Digistump Oak, WiFiduino, Amperka WiFi Slot, Seeed Wio Link, ESPectro Core, Schirmilabs Eduino WiFi, ITEAD Sonoff, DOIT ESP-Mx DevKit (ESP8285).

[Online Help](#)

[More Info](#)

Seleccione versión ▾

Instalar

Eliminar

## ESP Async E1.31

by [forkineye](#)

**Async E1.31 sACN for ESP8266.** Library for the asynchronous processing of sACN (E1.31 DMX over Ethernet) data.

[More info](#)

## ESPAsync\_WiFiManager

by [Khoi Hoang](#)

**ESP32 (including ESP32-S2 and ESP32-C3), ESP8266 WiFi Connection Manager using AsyncWebServer, with enhanced GUI and fallback Web ConfigPortal.** This Library is used for configuring ESP32 (including ESP32-S2 and ESP32-C3), ESP8266 modules WiFi Credentials at runtime. You can also specify static DNS servers, personalized HostName, fixed or random AP channel. Now with MultiWiFi auto(Re)connect, configurable CORS Header and auto-Timezone features.

[More info](#)

## ESPAsync\_WiFiManager\_Lite

by [Khoi Hoang](#)

**Light-Weight MultiWiFi/Credentials Async WiFiManager for ESP32 (including ESP32-S2 and ESP32-C3) and ESP8266 boards. Powerful-yet-simple-to-use feature to enable adding dynamic custom parameters.** Library using AsyncWebServer to configure MultiWiFi/Credentials at runtime for ESP32 (including ESP32-S2 and ESP32-C3) and ESP8266 boards. You can also specify DHCP HostName, static AP and STA IP. Use much less memory compared to full-fledge WiFiManager. Config Portal will be auto-adjusted to match the number of dynamic custom parameters. Optional default Credentials to be autoloading into Config Portal to use or change instead of manually input. Credentials are saved in LittleFS, SPIFFS or EEPROM. New powerful-yet-simple-to-use feature to enable adding dynamic custom parameters from sketch and input using the same Config Portal. Double or MultiDetectDetector as well as Virtual Switches feature permits entering Config Portal as requested. Configurable Customs HTML Headers, including Customs Style, Customs Head Elements, CORS Header.

[More info](#)

# Definición de librerías

```
#include <ESP8266WiFi.h>
```

```
#include <ESPAsyncTCP.h>
```

```
#include <ESPAsyncWebServer.h>
```

# Código

```
#include <ESP8266WiFi.h>
```

```
#include <ESPAsyncTCP.h>
```

```
#include <ESPAsyncWebServer.h>
```

```
const char* ssid = "Nombre de la red";
```

```
const char* password = "Contraseña de la red";
```

```
const int led_pin = 4;
```

```
String slider_value = "0";
```

```
const char* input_parameter = "value";
```

```
AsyncWebServer server(80);
```

```
const char index_html[] PROGMEM = R"rawliteral(
```

```
<!DOCTYPE HTML><html>
```

```
<head>
```

```
  <meta name="viewport" content="width=device-width,  
  initial-scale=1">
```

```
  <title>ESP8266 Control De Brillo Servidor Web</title>
```

<style>

html {font-family: Times New Roman; display: inline-block;  
text-align: center;}

h2 {font-size: 2.3rem;}

p {font-size: 2.0rem;}

body {max-width: 450px; margin:0px auto; padding-bottom:  
25px;}

.slider { -webkit-appearance: none; margin: 14px; width:  
360px; height: 25px; background: #FF0000;

outline: none; -webkit-transition: .2s; transition: opacity .2s;}

.slider::-webkit-slider-thumb {-webkit-appearance: none;  
appearance: none; width: 35px; height: 35px;  
background:#01070a; cursor: pointer;}

.slider::-moz-range-thumb { width: 35px; height: 35px;  
background: #01070a; cursor: pointer; }

</style>

</head>

<body>

<h2>ESP8266 Control De Brillo Servidor Web</h2>

<p><span id="textslider\_value">%SLIDERVALUE%</span></p>

```
<p><input type="range" onchange="updateSliderPWM(this)"
id="pwmSlider" min="0" max="255" value="%SLIDERVALUE%"
step="1" class="slider"></p>
```

```
<script>
```

```
function updateSliderPWM(element) {
```

```
    var slider_value =
document.getElementById("pwmSlider").value;
```

```
    document.getElementById("textslider_value").innerHTML =
slider_value;
```

```
    console.log(slider_value);
```

```
    var xhr = new XMLHttpRequest();
```

```
    xhr.open("GET", "/slider?value="+slider_value, true);
```

```
    xhr.send();
```

```
}
```

```
</script>
```

```
</body>
```

```
</html>
```

```
)rawliteral";
```

```
String processor(const String& var){
```

```
    if (var == "SLIDERVALUE"){
```

```
    return slider_value;
}
return String();
}
```

```
void setup(){
    Serial.begin(115200);

    WiFi.begin(ssid, password);
    while (WiFi.status() != WL_CONNECTED) {
        delay(1000);
        Serial.println("Connecting...");
    }
```

```
    Serial.println(WiFi.localIP());
```

```
    server.on("/", HTTP_GET, [](AsyncWebServerRequest
*request){
        request->send_P(200, "text/html", index_html, processor);
    });
```

```
server.on("/slider", HTTP_GET, [] (AsyncWebServerRequest
*request) {
    String message;
    if (request->hasParam(input_parameter)) {
        message = request->getParam(input_parameter)->value();
        slider_value = message;
        analogWrite(led_pin,slider_value.toInt());
    }
    else {
        message = "No message sent";
    }
    Serial.println(message);
    request->send(200, "text/plain", "OK");
});
```

```
server.begin();
}
```

```
void loop() {
```

}