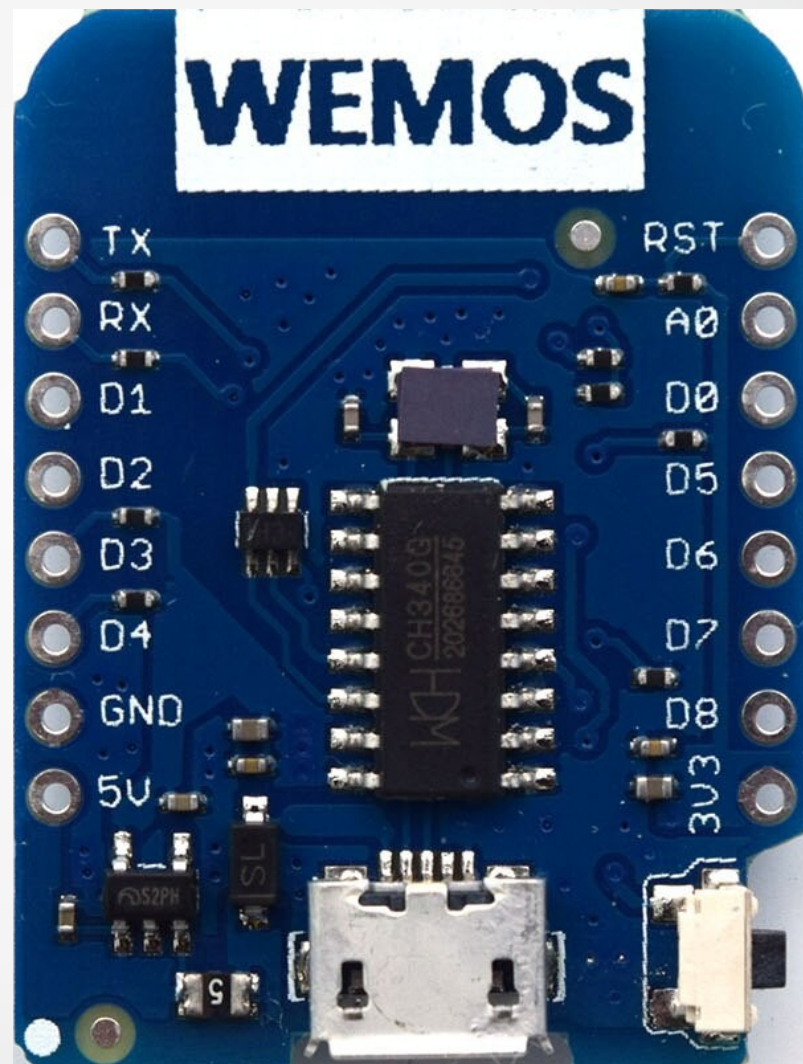
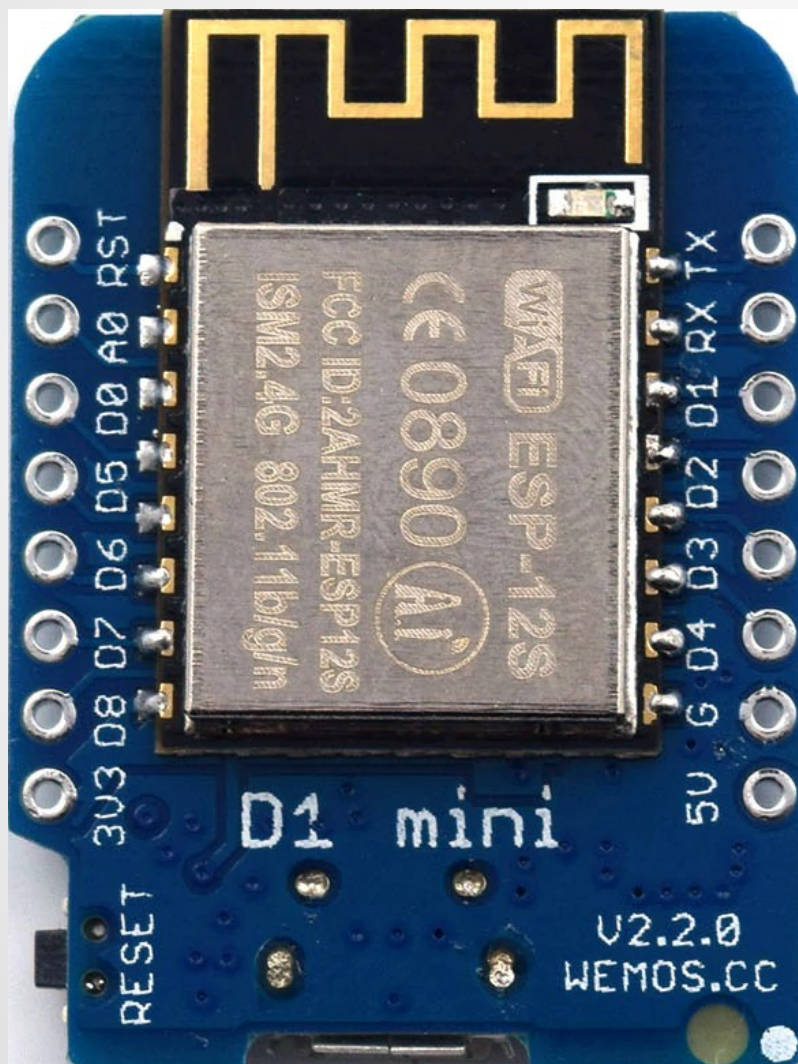
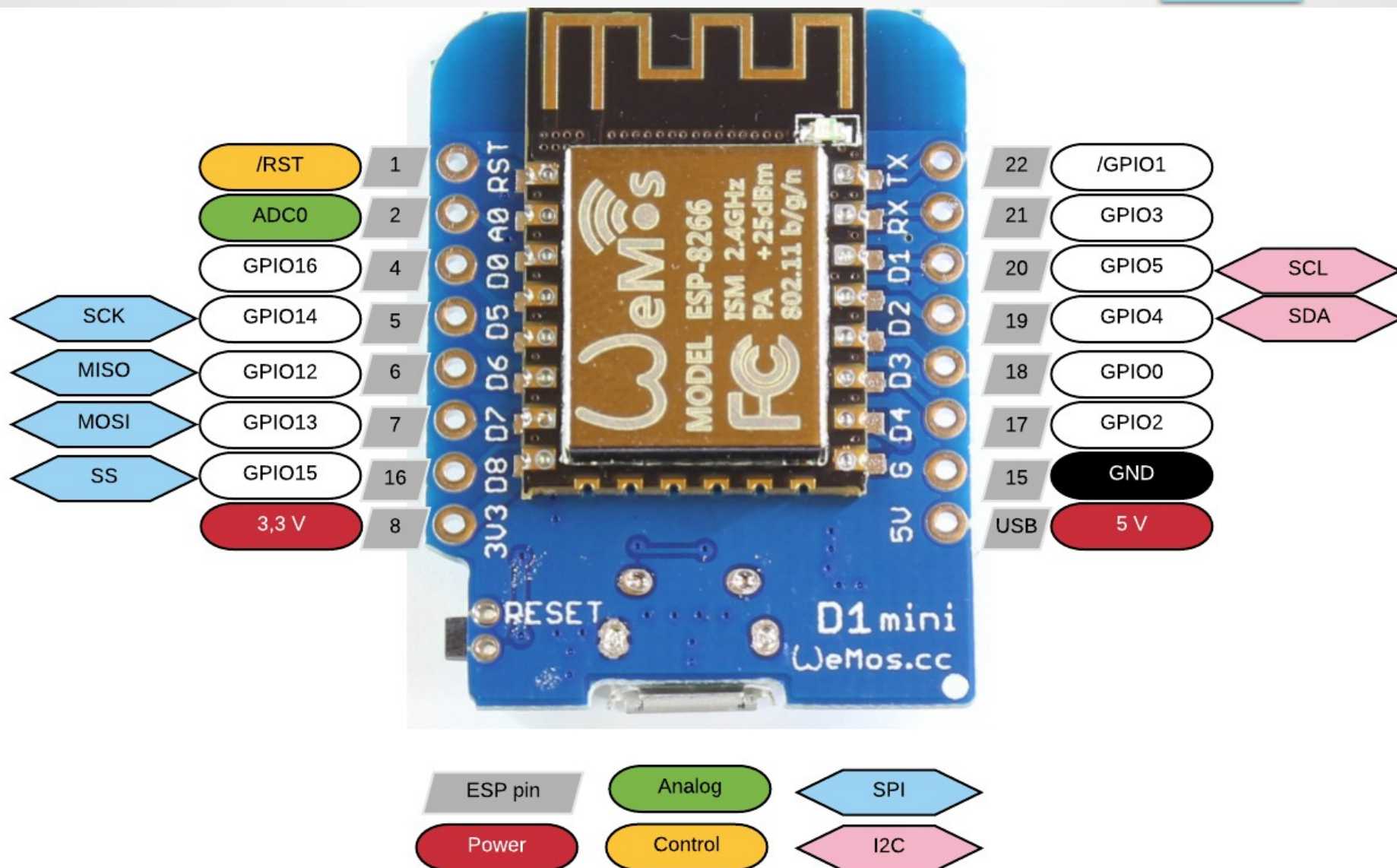


# Introducción - Programar ESP8266



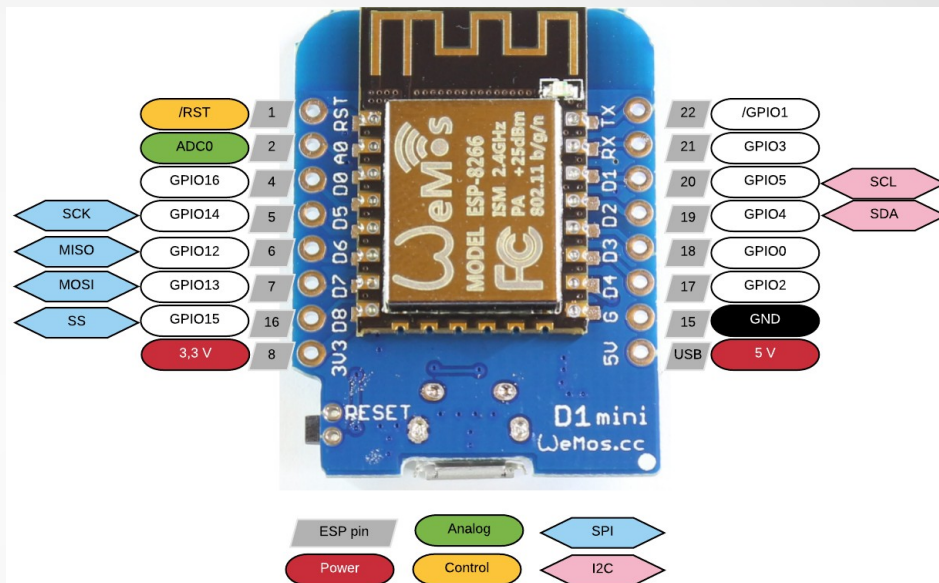
# Introducción – pinout Wemos D1 mini





# Introducción – pinout Wemos D1 mini

Pin	ESP-8266 Pin	Función
TX	TXD	TXD
RX	RXD	RXD
A0	A0	Analog input (max 3.2V)
D0	GPIO16	IO
D1	GPIO5	IO, PWM, Interrupt, I2C, SCL
D2	GPIO4	IO, PWM, Interrupt, I2C, SDA
D3	GPIO0	IO 10k Pull-up, PWM, Interrupt, I2C
D4	GPIO2	IO 10k Pull-up, PWM, Interrupt, I2C, BUILTIN_LED
D5	GPIO14	IO, PWM, Interrupt, I2C, SCK
D6	GPIO12	IO, PWM, Interrupt, I2C,, MISO
D7	GPIO13	IO, PWM, Interrupt, I2C,, MOSI
D8	GPIO15	IO 10k Pull-down, PWM, Interrupt, I2C,, SS
G	GND	Ground
5V	-	5V
3V3	3.3V	3.3V
RST	RST	Reset



# Introducción – Configurar IDE de Arduino

- Para poder comunicar o ESP8266 co IDE de Arduino, é preciso que este recoñeza a placa. Para isto debemos ir ao menú de configuración e indicarlle ao entorno, a url en que pode encontrar a descrición das placas da familia:

[http://arduino.esp8266.com/stable/  
package\\_esp8266com\\_index.json](http://arduino.esp8266.com/stable/package_esp8266com_index.json)

- A continuación temos que ir ao menú ferramentas e no item placas, escoller o xestor de tarxetas. No buscador procuramos 'esp8266 by esp8266 community' e dámoslle a instalar.
- Xa temos dispoñibles as placas da familia do ESP8266, a que nós imos usar é 'Generic ESP8266 Module'



# Introducción – Configurar IDE de Arduino

Preferências

Configurações Rede

Localização do bloco de rascunhos:

Linguagem do editor:  (requer reinício do Arduino)

Tamanho da Fonte do Editor:

Escala da interface: ☒ Automático  % (requer reinício do Arduino)

Tema:  (requer reinício do Arduino)

Mostrar mensagens detalhadas durante: ☐ compilação ☐ upload

Avisos do compilador:

☒ Mostrar numeros de linha ☐ Activar Quebra de Linhas no Código

☒ Verificar o código após o envio ☐ Usar um editor externo

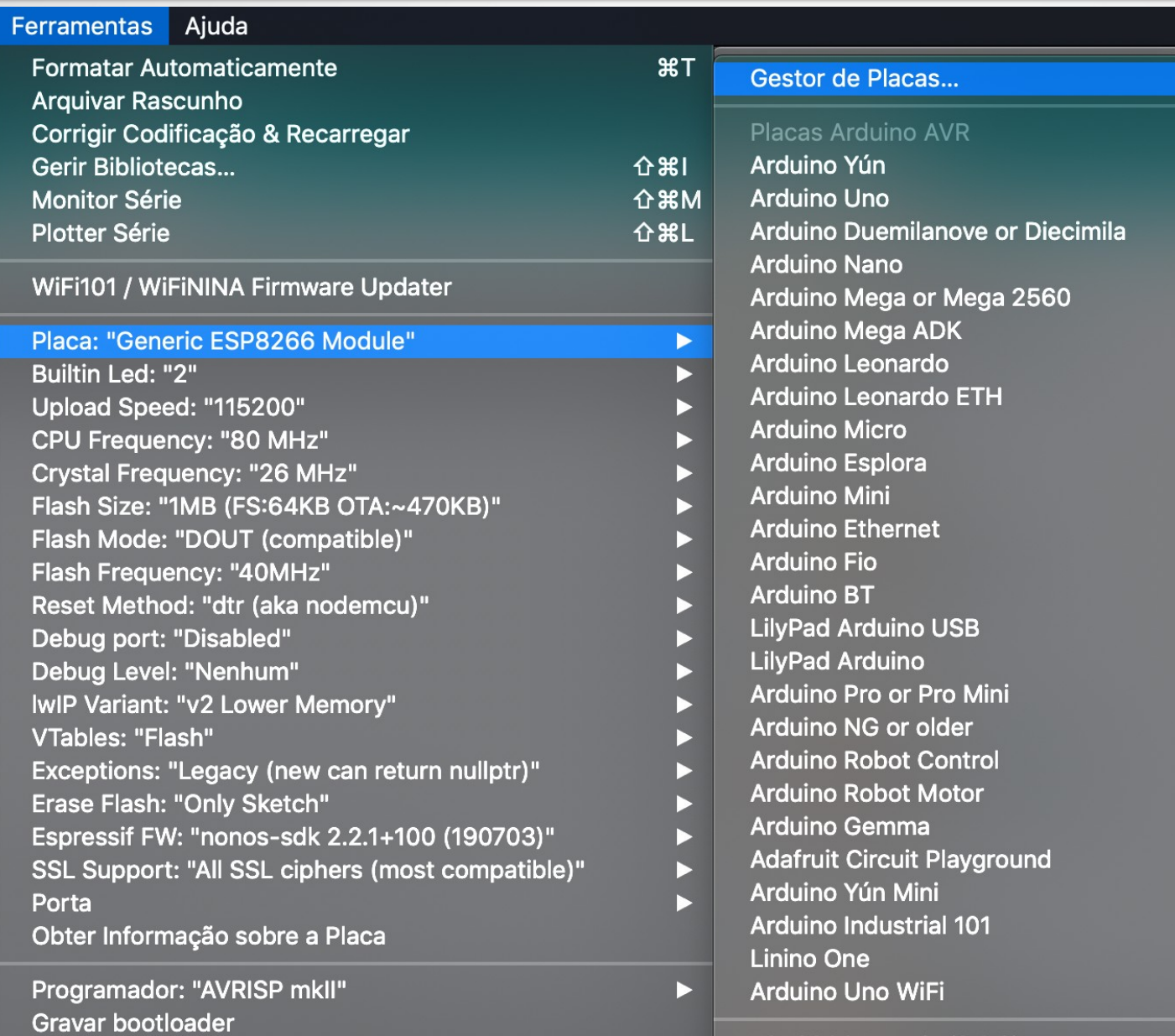
☒ Procurar por atualizações ao iniciar ☒ Guardar enquanto verifica ou envia

☒ Use accessibility features

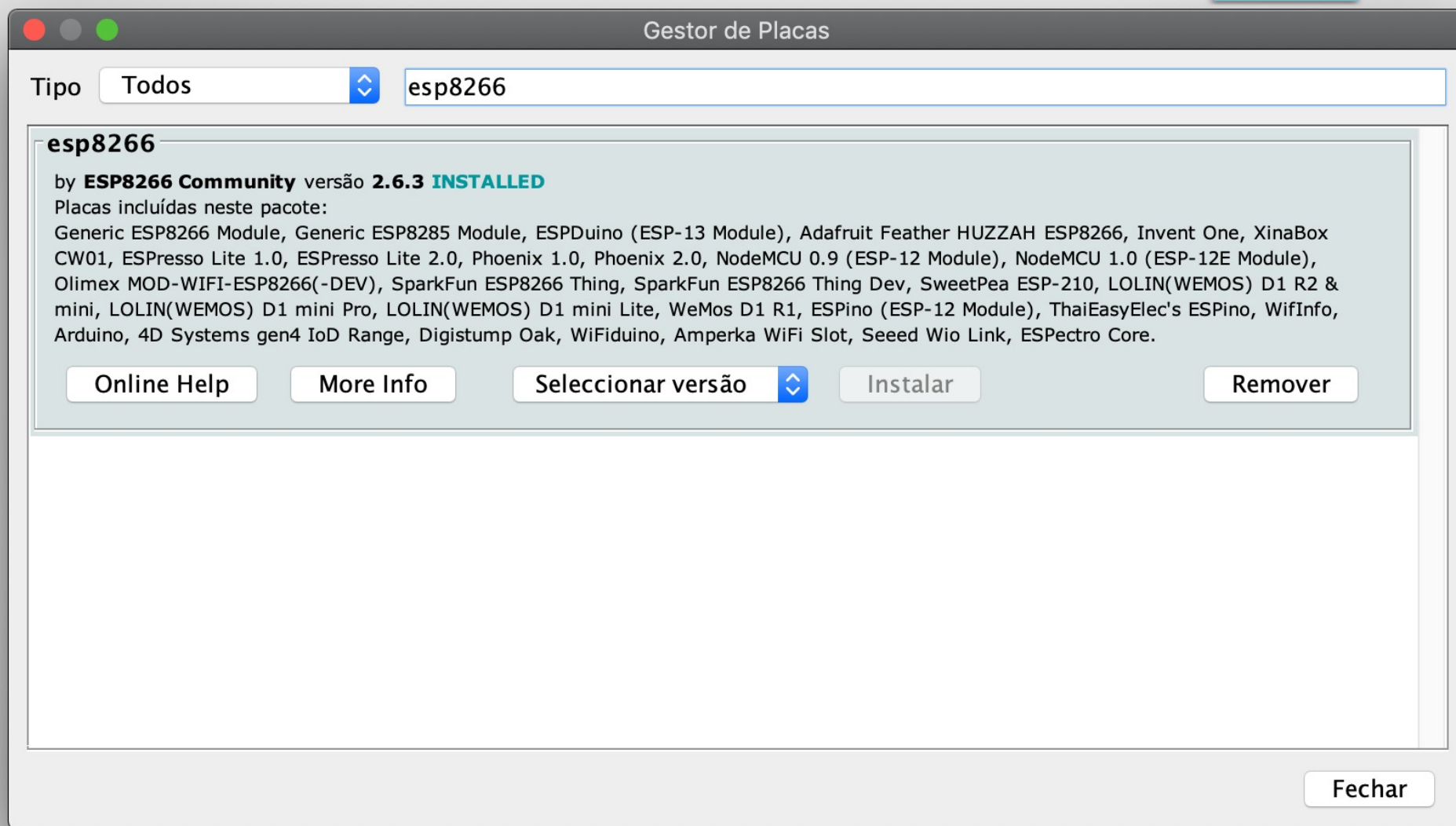
URL Adicionais do Gestor de Placas:

Outras preferências podem ser alteradas diretamente no ficheiro  
`/Users/mercurio/Library/Arduino15/preferences.txt`  
(altere apenas quando o Arduino não estiver em execução)

# Introducción – Configurar IDE de Arduino



# Introducción – Configurar IDE de Arduino



ARDUINO



# Introducción – Configurar IDE de Arduino

Ferramentas	Ajuda
Formatar Automaticamente	⌘T
Arquivar Rascunho	
Corrigir Codificação & Recarregar	
Gerir Bibliotecas...	⇧⌘I
Monitor Série	⇧⌘M
Plotter Série	⇧⌘L
WiFi101 / WiFinINA Firmware Updater	
Placa: "Generic ESP8266 Module" ▶	
Built-in Led: "2"	▶
Upload Speed: "115200"	▶
CPU Frequency: "80 MHz"	▶
Crystal Frequency: "26 MHz"	▶
Flash Size: "1MB (FS:64KB OTA:~470KB)"	▶
Flash Mode: "DOUT (compatible)"	▶
Flash Frequency: "40MHz"	▶
Reset Method: "dtr (aka nodemcu)"	▶
Debug port: "Disabled"	▶
Debug Level: "Nenhum"	▶
lwIP Variant: "v2 Lower Memory"	▶
VTables: "Flash"	▶
Exceptions: "Legacy (new can return nullptr)"	▶
Erase Flash: "Only Sketch"	▶
Espressif FW: "nonos-sdk 2.2.1+100 (190703)"	▶
SSL Support: "All SSL ciphers (most compatible)"	▶
Porta	▶
Obter Informação sobre a Placa	
Programador: "AVRISP mkII"	▶
Gravar bootloader	

ESP8266 Boards (2.6.3)

- ✓ Generic ESP8266 Module
- Generic ESP8285 Module
- 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 Pro
- LOLIN(WEMOS) D1 mini Lite
- 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
- ITEAD Sonoff
- DOIT ESP-Mx DevKit (ESP8285)





# Introducción – Configurar IDE de Arduino

- Unha vez escollida a placa, podemos probar a subir scripts xa elaborados, que están dispoñibles no menú  
    ‘Ficheiro > Exemplos > Exemplos para Generic ESP8266 Module’
- Un clásico é cargar ‘ESP8266 > Blink’, que é como o ‘Ola mundo’ do HW libre.
- Nós imos realizar o noso propio blink, conectando un LED físico no pin 14.
- No Wemos D1 mini, pasa algo parecido que coa placa Arduino UNO: o LED azul que está fisicamente unido ao pin TX (GPIO01) pódese usar para probas co parámetro predefinido ‘LED\_BUILTIN’.
- Para evitar interferencias coa transmisión serie, imos empregar outro pin diferente.



# Saídas dixitais – Encender un LED

led.esp8266 | Arduino 1.8.12

✓


↺







Envio Usando o Programador



led.esp8266

```
1 /*
2  * Encendido e apagado dun LED no pin D5
3  * da placa ESP8266.
4  *
5  */
6
7 // GPIO14: etiquetado en placa como D5
8 #define LED 14
9
10 // Tempo de espera
11 int tempo = 1000;
12
13 void setup() {
14     Serial.begin(9600);
15     pinMode(LED, OUTPUT);
16 }
17
18 void loop() {
19     digitalWrite(LED, LOW);
20     Serial.println("LED off");
21     delay(tempo);
22     digitalWrite(LED, HIGH);
23     Serial.println("LED on");
24     delay(tempo);
25 }
```

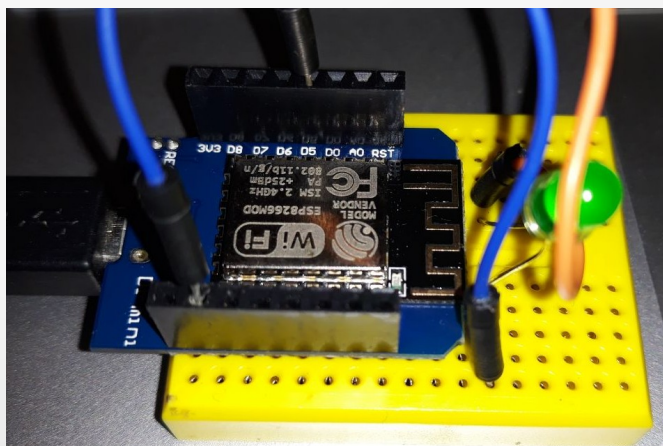
Carregamento completo

Leaving...  
Hard resetting via RTS pin...

[illegible]☒ Avanço automático de linha ☐ Mostra

O script é similar o que xa temos feito para o Arduino UNO, coa diferenza de que temos que seleccionar outra placa distinta.

Para o noso HW particular a placa a escoller é 'Generic ESP8266 Module'.



# Introducción – Configurar IDE de Arduino

- Nesta unidade aprendemos a:
  - recoñecer o pinout dun Wemos D1 mini
  - configurar o IDE de Arduino para comunicarse co diversas placas da familia do ESP8265 e programalo empregando de xeito similar ao Arduino UNO
  - conectar un LED a unha saída dixital e comprobar o funcionamento da placa
  - comprobar a saída serie mediante mensaxes do script