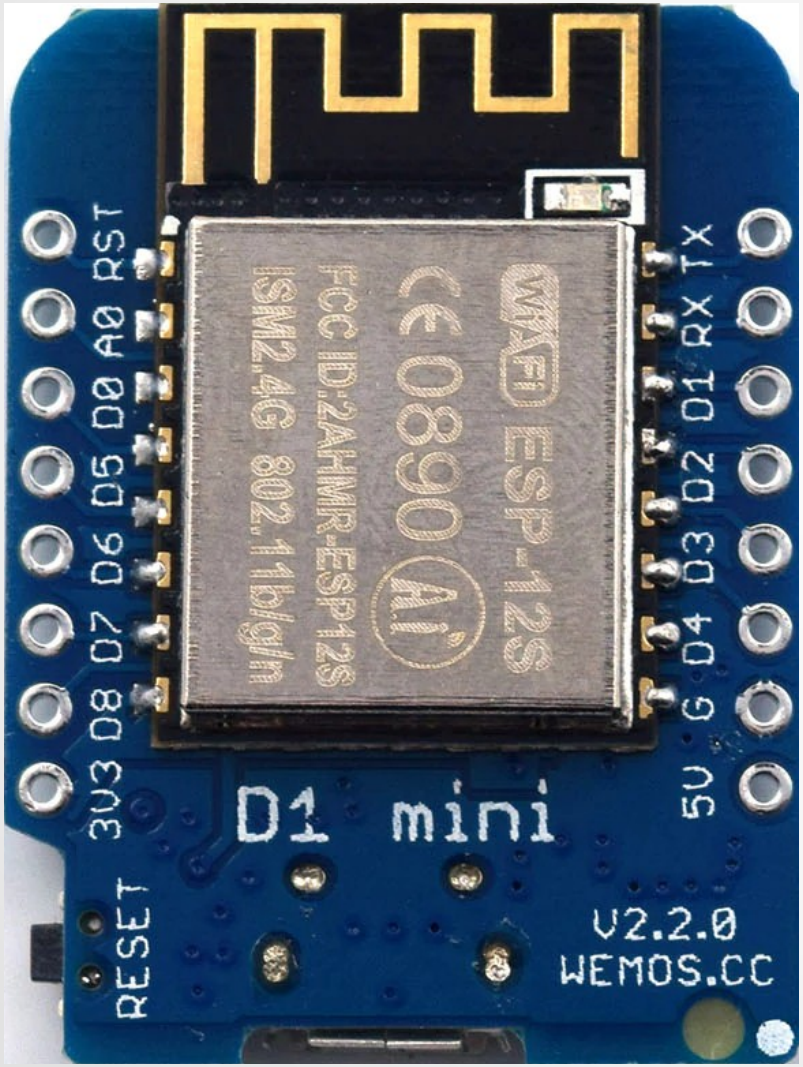
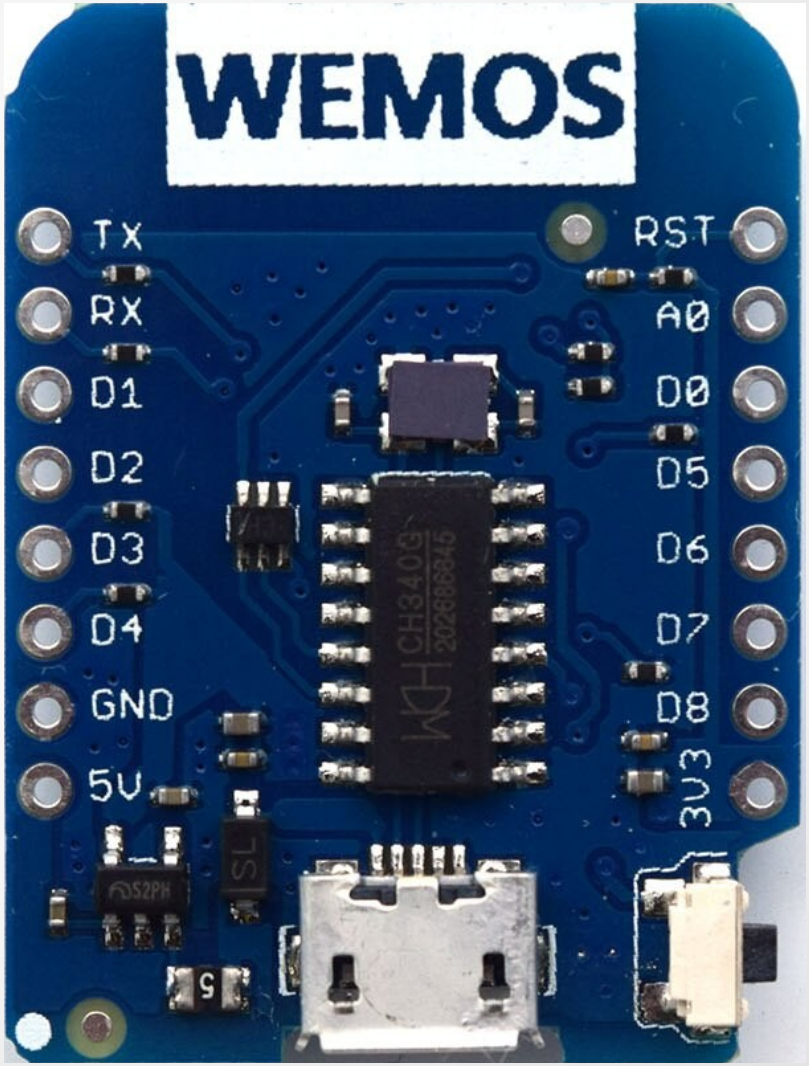
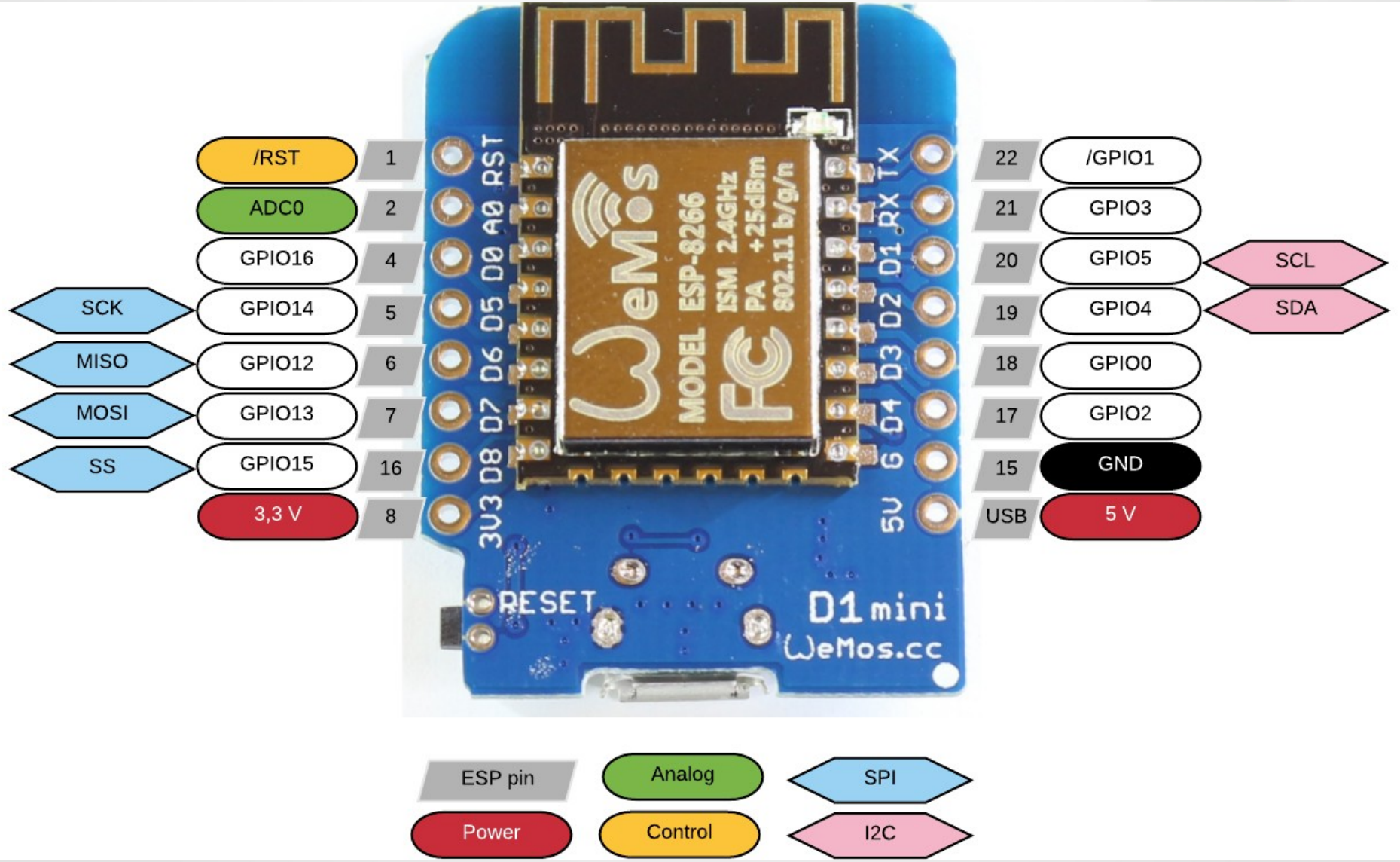
ESP8266 – Conexión a WiFi existente





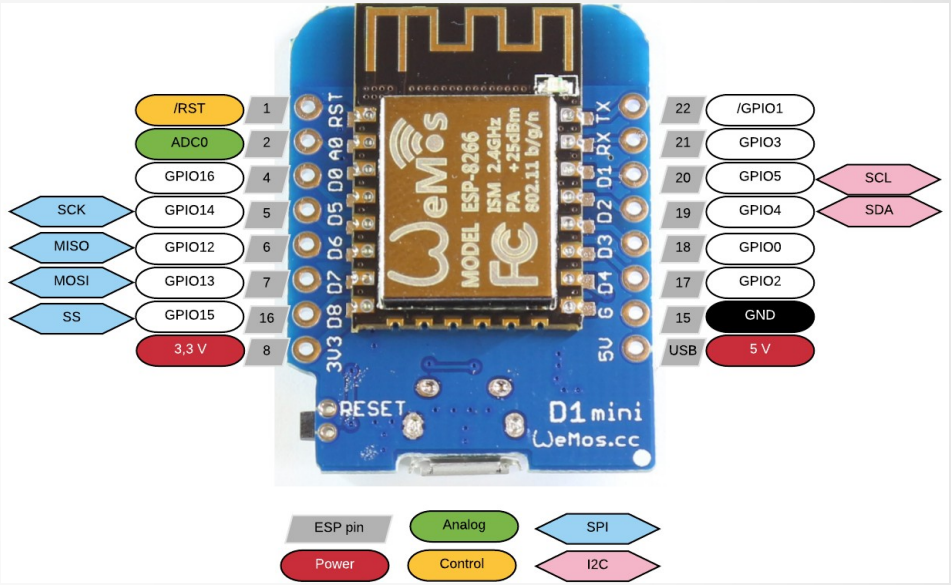
Víctor M. Álvarez 202003

Pinout Wemos D1 mini



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



ESP8266 – Conexión a WiFi existente

- A placa ESP8266 inclúe un módulo WiFi que nos permite comunicalo con outros elementos da rede.
- Existen diversas librerías que nos permiten usar con certa facilidade as funcións de rede do módulo WiFi. Nós empregaremos a librería:

<ESP8266WiFi.h>

- Podemos configurar a placa como:

WIFI_STA	Modo estación, conéctase a unha rede existente previamente configurada
WIFI_AP	Modo Access Point, xenera unha WiFi á que se poden conectar outros dispositivos
WIFI_AP_STA	Combinación de modo AP e STA
WIFI_OFF	Apagada

Víctor . Álvarez 201911

ESP8266 – Conexión a WiFi existente

- A librería ESP8266WiFi forma parte da definición de placas que descargamos co xestor de placas.
- Para usala, temos que empregar o obxecto WiFi, que representará un interface de rede. Á súa vez proporciónanos diversas funcións (métodos), das que imos empregar polo de agora:
  - WiFi.mode(WIFI_STA) - configuramos o dispositivo para conectarse a unha WiFi preexistente
  - WiFi.disconnect() - desconectamos da rede actual
  - WiFi.begin(SSID, PASS) – conectamos a unha rede co nome SSID e chave PASS
  - WiFi.localIP() - devolve a IP configurada unha vez conectado á rede
  - WiFi.Status() - obtemos en que estado está a conexión á rede. Os estados posibles aparecen na diapositiva seguinte

Víctor . Álvarez 201911

ESP8266 – Conexión a WiFi existente

- Estados posibles da conexión de rede:

WL_IDLE_STATUS	A WiFi está mudando entre estados
WL_NO_SSID_AVAIL	Non se encontra rede co identificador SSID
WL_CONNECT	Conexión realizada con éxito
WL_CONNECT_FAILED	Chave de rede incorrecta
WL-DISCONNECTED	O ESP8266 non está configurado en modo STA

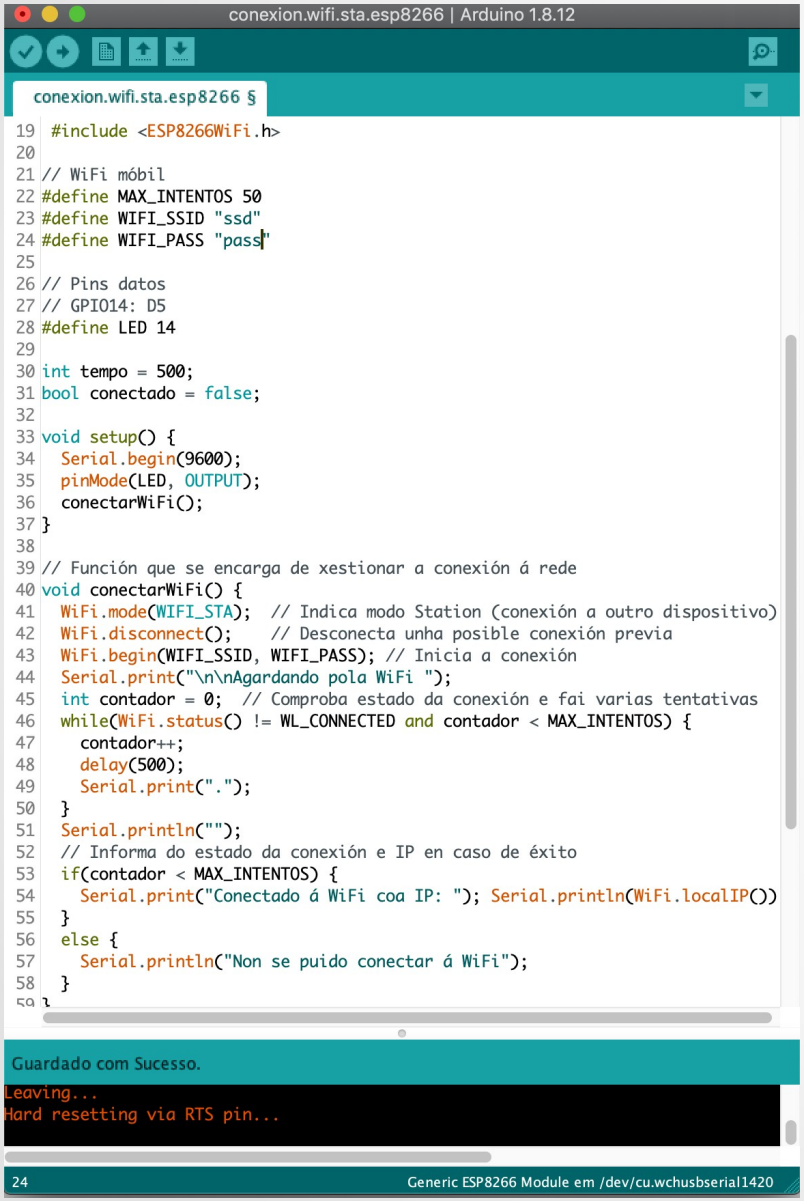
Víctor . Álvarez 201911

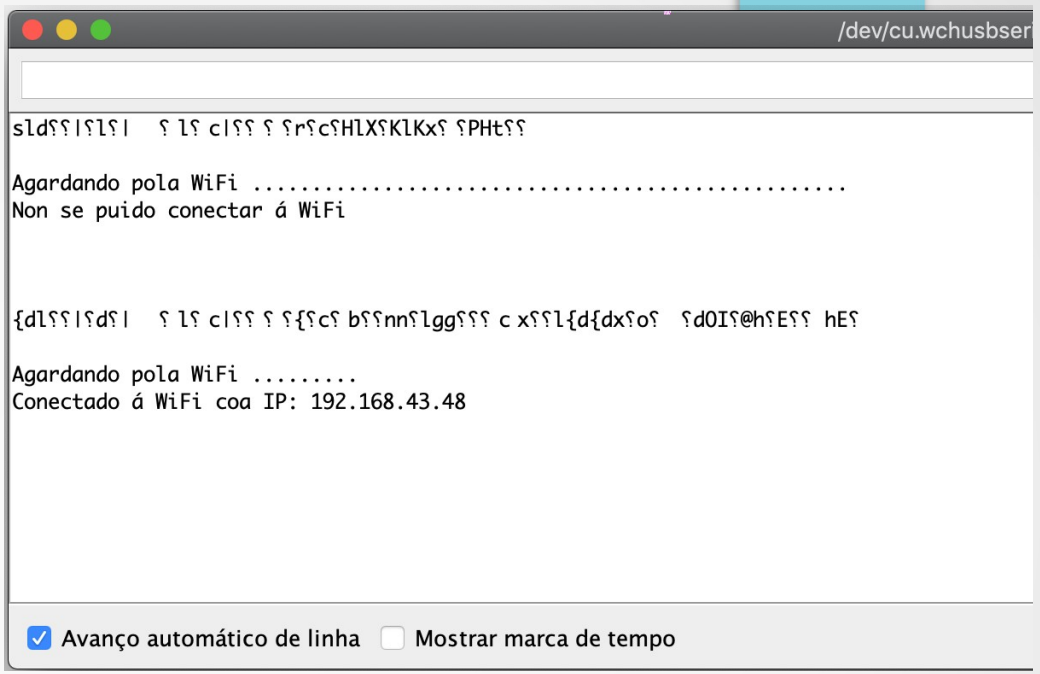
ESP8266 – Conexión a WiFi existente

- Imos empezar por conectar o Wemos D1 a unha WiFi creada para a ocasión nun móbil.
- O script que executará o Wemos tentará conectarse á rede (dispoñendo do nome e chave) durante un número determinado de intentos. Se ao finalizar o número de intentos non ten éxito, comunicará o feito polo porto serie. En caso de conectarse con éxito, mostrará polo porto serie a IP coa que se conecta á rede.
- Para aproveitar o código do anterior script, imos facer que o LED escintile rápido cando hai erro de rede e que escintile máis lento cando hai conexión de rede.

Víctor . Álvarez 201911

ESP8266 – Conexión a WiFi existente





- O script completo está na carpeta correspondente dos scripts do repositorio, en Github.

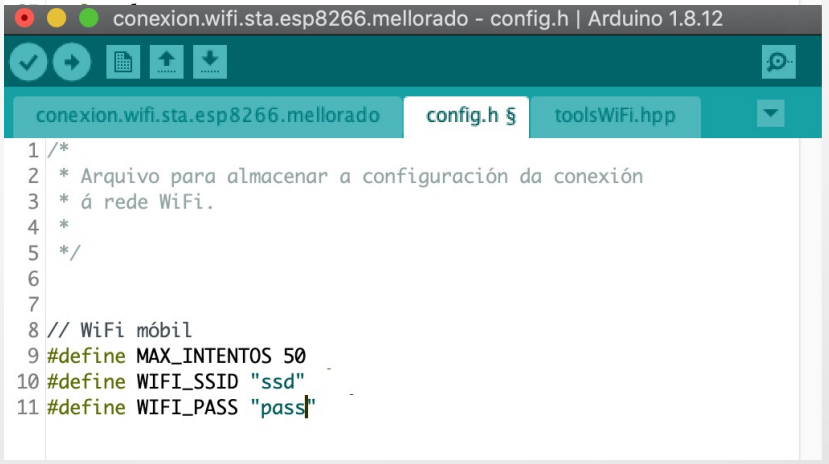
Víctor . Álvarez 201911

ESP8266 – Conexión a WiFi existente

- Imos mellorar un pouco o script anterior, no sentido de crear un arquivo de configuración ‘config.h’, no que imos gardar o nome de rede e a chave, para que non sexa público nos repositorios e máis adiante outras variables necesarias para o noso script.
- A función conectarWiFi() tamén a imos pasar a outro arquivo ‘toolsWiFi.hpp’, que actuará como unha librería creada por nós e onde gardaremos as funcións relacionadas coa rede, para dispor dun código principal máis limpo.
- Para crear estes arquivos, podemos facelo polo método habitual ou creando arquivos de texto plano, ou mellor aínda premendo do triángulo co vértice cara abaixo do IDE de Arduíno. Aparece un menú contextual, no que podemos crear un novo ‘separador’, que non é máis que un novo arquivo de texto (coa extensión que nós queiramos) dentro da carpeta do noso proxecto.
- É interesante ademais que estes separadores van seguir unidos á xanela aínda que pechemos o IDE, formando parte do noso código.

Víctor . Álvarez 201911

ESP8266 – Conexión a WiFi existente



- O código principal queda máis claro e compacto.
- O código secundario queda nos outros dous ‘separadores’.
- Impórtase o código secundario co nome do arquivo entre comillas dobles (“ e “), para indicar que son locais.

Víctor . Álvarez 201911

ESP8266 – Conexión a WiFi existente

- Outras funcións que nos poden ser útiles en proxectos que impliquen WiFi co ESP8266 son:
  - WiFi.reconnect() - Tenta a reconexión cando se cancela por sinal baixo ou tempo de inactividade
  - WiFi.disconnect(true) – Desconecta a rede cando sexa necesario aforrar batería
  - WiFi.isConnected() - Para empregar en esquemas condicionais (if-else, switch-case)
  - WiFi.setAutoConnect(autoConnect) – Tena a reconexión sempre que se desconecte
  - WiFi.status() - Informa do esttado de conexión da red
- Se precisamos coñecer datos da nosa conexión: nome de rede (en caso de multirrede), IP, porta de enlace, etc, as funcións que nos dan esa informacion son:
  - WiFi.SSID()
  - WiFi.hostname()
  - WiFi.localIP()
  - WkFi.subnetMask()
  - WiFi.gatewayIP()
  - WiFi.dnsIP(dns_no)
  - WiFi.macAddress()

Víctor . Álvarez 201911

ESP8266 – Conexión a WiFi existente

- Nesta unidade aprendemos a:
  - recoñecer os modos de rede (STA ou AP)
  - recoñecer os posibles estados da conexión
  - conectarnos a unha rede existente e consultar parámetros da mesma
  - mellorar o código separándoo en librerías definidas polo usuario (locais), ou eliminar información sensible trasladándoa a arquivos de configuración

Víctor . Álvarez 201911