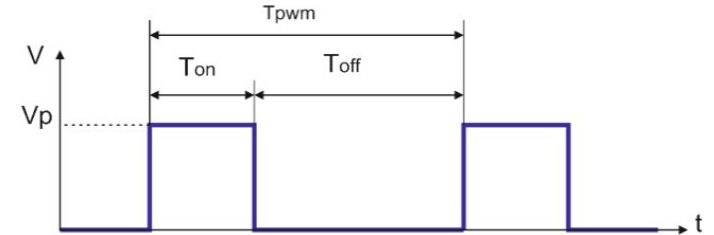


# Agenda Clase II

<b>09:30 hs - Enlace de conexión</b>	Consultas ejercicios clase 1
<b>10:00 hs - Conceptos</b>	Comunicación UART. Ejemplos transmisión y recepción. Ejemplo de conexión WiFi. Ciclos FOR y WHILE.
<b>10:40 hs - Programación embebida</b>	Salida digital PWM con funciones de librería. Ejemplo integrador. Sonda de T° DS18B20.
<b>11:15 hs - Break</b>	Receso 15 minutos.
<b>11:30 hs - Cloud</b>	Plataformas IoT. Servidor Blynk. Generación de interfaz en Blynk. Vínculo con programa en ESP8266.
<b>12:45 hs - Ejercitación</b>	Espacio de consultas. Próximos pasos.

# Conceptos anteriores - Ejercicios CLASE 1

- Salidas Digitales  $\rightarrow$  PWM
- Entradas Digitales
- Entradas Analógicas



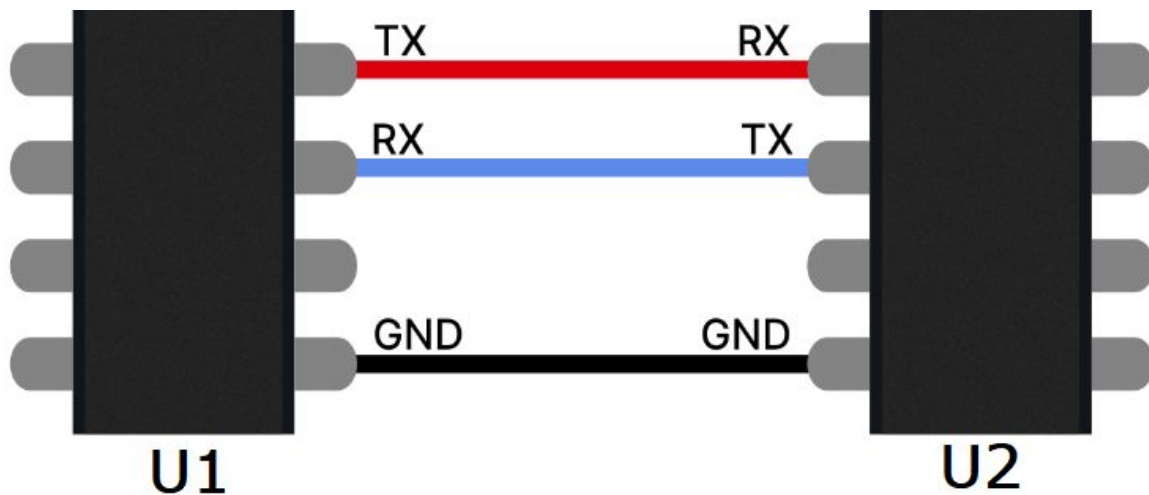
*¿Alguna duda?*

## *Clase II - Módulo A*

# Más conceptos y Programación Embebida

## UART

- UNIVERSAL
- ASÍNCRONA
- RECEPCIÓN
- TRANSMISIÓN



## UART

### PARÁMETROS

- Baudrate
- Bits de DATOS
- Bits de STOP
- Bits de paridad

### DISPOSITIVOS

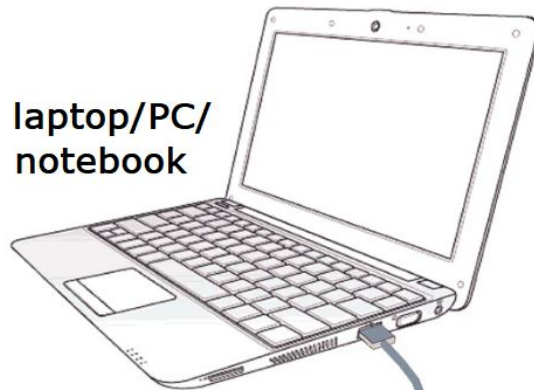
- Sólo 2

Serial port settings

Port configuration	Transmitted text	Options
Port: COM7	<input type="radio"/> Append nothing <input type="radio"/> Append CR <input type="radio"/> Append LF <input checked="" type="radio"/> Append CR-LF <input type="checkbox"/> Local echo	<input checked="" type="checkbox"/> Stay on top <input checked="" type="checkbox"/> Quit on Escape <input type="checkbox"/> Autocomplete edit line <input checked="" type="checkbox"/> Keep history <input type="checkbox"/> Close port when inactive
Baud rate: 115200	Received text Polling: 100 ms Max. lines: <input type="text"/> Font: default <input type="checkbox"/> Word wrap	Plug-ins <input type="checkbox"/> Auto Reply <input type="checkbox"/> Function Keys <input type="checkbox"/> Hex View <input type="checkbox"/> Highlight <input type="checkbox"/> Log File
Data bits: 8		
Stop bits: 1		
Parity: none		
Flow control: none		
Forward: none		

User interface language: English (en)

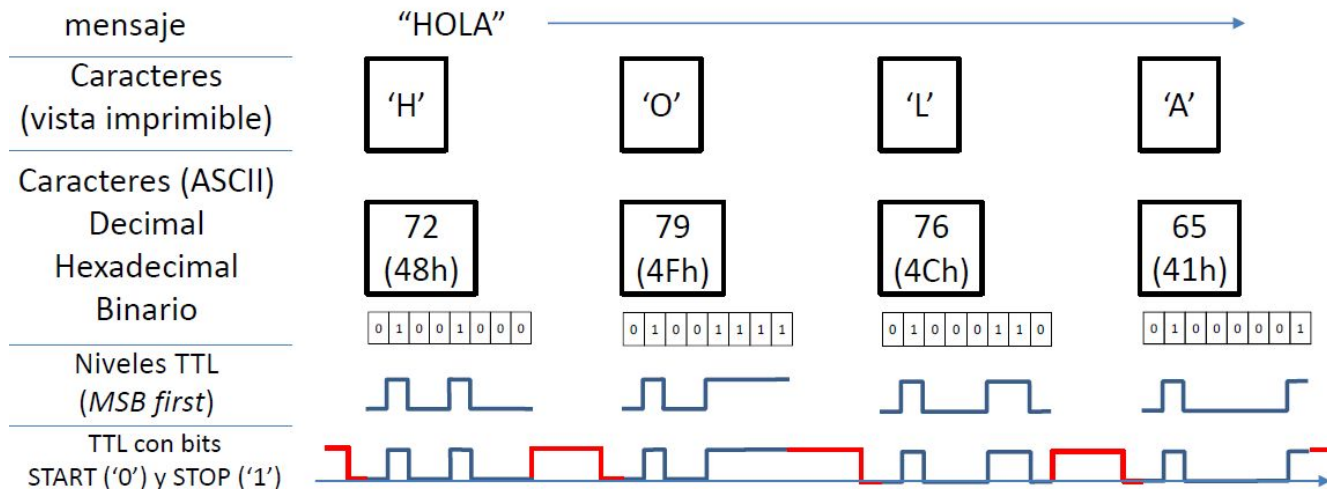
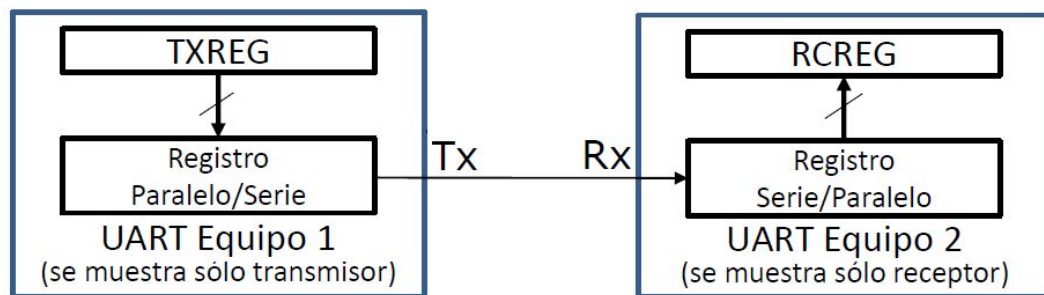
U2: laptop/PC/  
notebook



U1: ESP8266



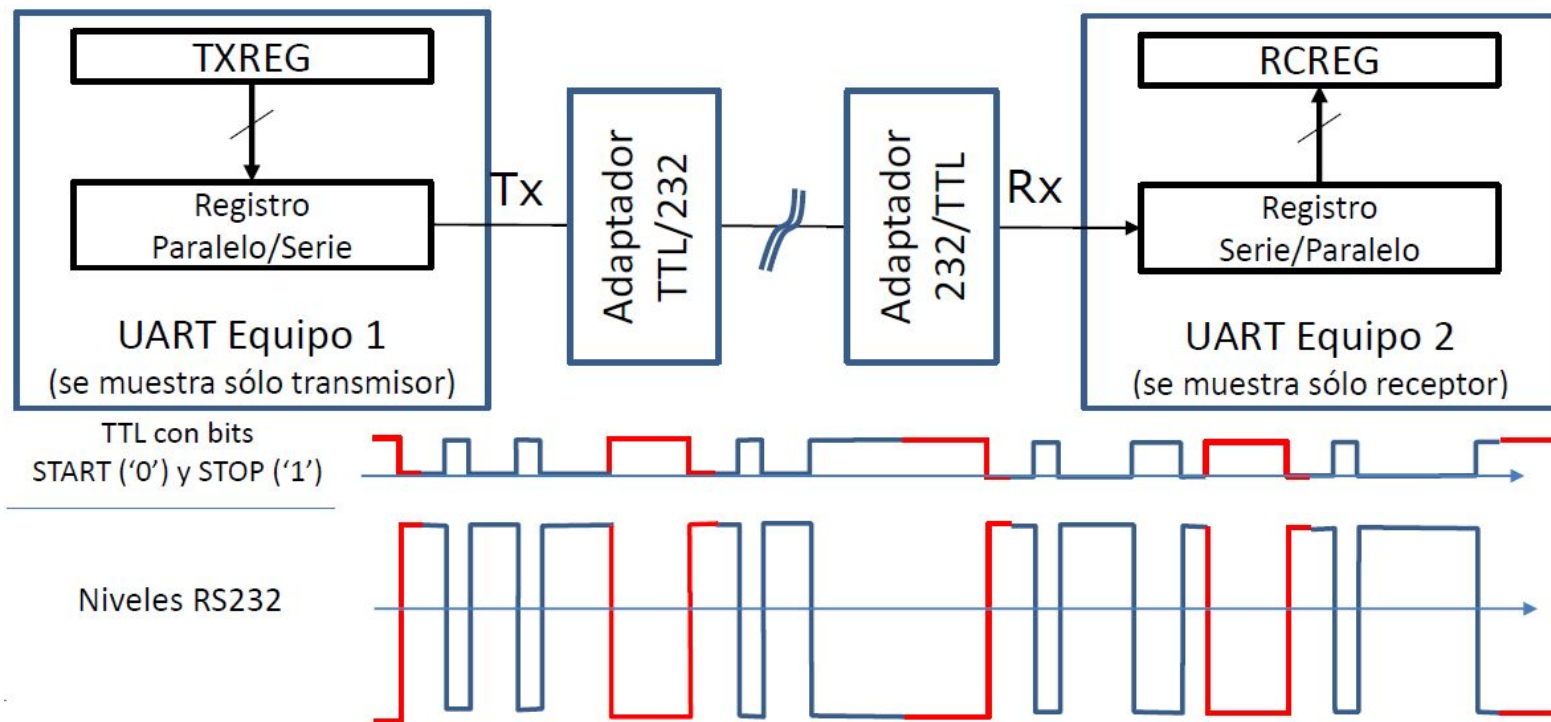
USB Cable



# ASCII

Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char	Decimal	Hexadecimal	Binary	Octal	Char
0	0	0	0	[NULL]	48	30	110000	60	0	96	60	1100000	140	`
1	1	1	1	[START OF HEADING]	49	31	110001	61	1	97	61	1100001	141	a
2	2	10	2	[START OF TEXT]	50	32	110010	62	2	98	62	1100010	142	b
3	3	11	3	[END OF TEXT]	51	33	110011	63	3	99	63	1100011	143	c
4	4	100	4	[END OF TRANSMISSION]	52	34	110100	64	4	100	64	1100100	144	d
5	5	101	5	[ENQUIRY]	53	35	110101	65	5	101	65	1100101	145	e
6	6	110	6	[ACKNOWLEDGE]	54	36	110110	66	6	102	66	1100110	146	f
7	7	111	7	[BELL]	55	37	110111	67	7	103	67	1100111	147	g
8	8	1000	10	[BACKSPACE]	56	38	111000	70	8	104	68	1101000	150	h
9	9	1001	11	[HORIZONTAL TAB]	57	39	111001	71	9	105	69	1101001	151	i
10	A	1010	12	[LINE FEED]	58	3A	111010	72	:	106	6A	1101010	152	j
11	B	1011	13	[VERTICAL TAB]	59	3B	111011	73	;	107	6B	1101011	153	k
12	C	1100	14	[FORM FEED]	60	3C	111100	74	<	108	6C	1101100	154	l
13	D	1101	15	[CARRIAGE RETURN]	61	3D	111101	75	=	109	6D	1101101	155	m
14	E	1110	16	[SHIFT OUT]	62	3E	111110	76	>	110	6E	1101110	156	n
15	F	1111	17	[SHIFT IN]	63	3F	111111	77	?	111	6F	1101111	157	o
16	10	10000	20	[DATA LINK ESCAPE]	64	40	1000000	100	@	112	70	1110000	160	p
17	11	10001	21	[DEVICE CONTROL 1]	65	41	1000001	101	A	113	71	1110001	161	q
18	12	10010	22	[DEVICE CONTROL 2]	66	42	1000010	102	B	114	72	1110010	162	r
19	13	10011	23	[DEVICE CONTROL 3]	67	43	1000011	103	C	115	73	1110011	163	s
20	14	10100	24	[DEVICE CONTROL 4]	68	44	1000100	104	D	116	74	1110100	164	t
21	15	10101	25	[NEGATIVE ACKNOWLEDGE]	69	45	1000101	105	E	117	75	1110101	165	u
22	16	10110	26	[SYNCHRONOUS IDLE]	70	46	1000110	106	F	118	76	1110110	166	v
23	17	10111	27	[END OF TRANS. BLOCK]	71	47	1000111	107	G	119	77	1110111	167	w
24	18	11000	30	[CANCEL]	72	48	1001000	110	H	120	78	1111000	170	x
25	19	11001	31	[END OF MEDIUM]	73	49	1001001	111	I	121	79	1111001	171	y
26	1A	11010	32	[SUBSTITUTE]	74	4A	1001010	112	J	122	7A	1111010	172	z
27	1B	11011	33	[ESCAPE]	75	4B	1001011	113	K	123	7B	1111011	173	{
28	1C	11100	34	[FILE SEPARATOR]	76	4C	1001100	114	L	124	7C	1111100	174	
29	1D	11101	35	[GROUP SEPARATOR]	77	4D	1001101	115	M	125	7D	1111101	175	}
30	1E	11110	36	[RECORD SEPARATOR]	78	4E	1001110	116	N	126	7E	1111110	176	~
31	1F	11111	37	[UNIT SEPARATOR]	79	4F	1001111	117	O	127	7F	1111111	177	[DEL]
32	20	100000	40	[SPACE]	80	50	1010000	120	P					
33	21	100001	41	!	81	51	1010001	121	Q					
34	22	100010	42	"	82	52	1010010	122	R					
35	23	100011	43	#	83	53	1010011	123	S					
36	24	100100	44	\$	84	54	1010100	124	T					
37	25	100101	45	%	85	55	1010101	125	U					
38	26	100110	46	&	86	56	1010110	126	V					
39	27	100111	47	'	87	57	1010111	127	W					
40	28	101000	50	(	88	58	1011000	130	X					
41	29	101001	51	)	89	59	1011001	131	Y					
42	2A	101010	52	*	90	5A	1011010	132	Z					
43	2B	101011	53	+	91	5B	1011011	133	[					
44	2C	101100	54	,	92	5C	1011100	134	\					
45	2D	101101	55	-	93	5D	1011101	135	]					
46	2E	101110	56	.	94	5E	1011110	136	^					
47	2F	101111	57	/	95	5F	1011111	137	_					

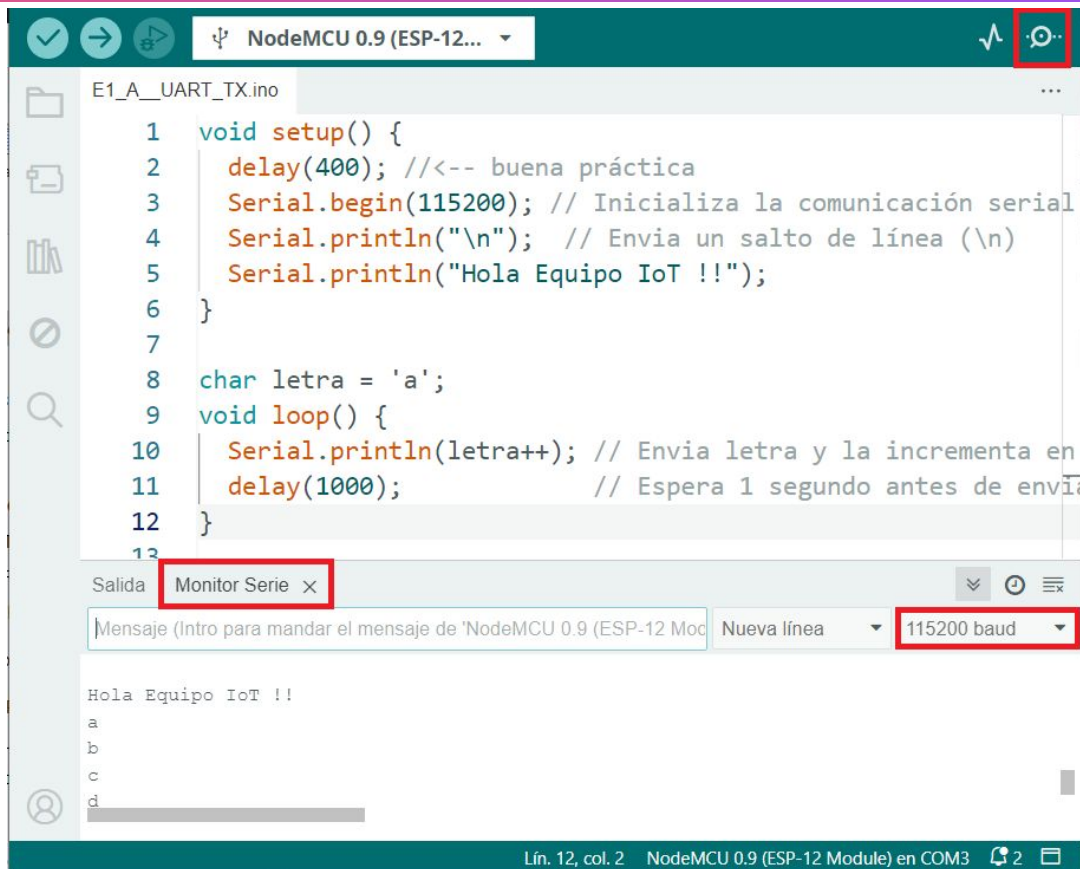




## *Ejemplo Transmisión*

```
void setup() {  
    delay(400); //<-- buena práctica  
    Serial.begin(115200); // Inicializa la comunicación serial a 115200 baudios  
    Serial.println("\n"); // Envía un salto de línea (\n)  
    Serial.println("Hola Equipo IoT !!");  
}  
  
char letra = 'a';  
void loop() {  
    Serial.println(letra++); // Envía letra y la incrementa en 1 en cada iteración  
    delay(1000);             // Espera 1 segundo antes de enviar el siguiente mensaje  
}
```

## Ejemplo Transmisión



The screenshot shows the Arduino IDE interface. The top toolbar includes icons for checking, running, and uploading code, along with a dropdown menu for the board (NodeMCU 0.9 (ESP-12...)). The main editor displays the code for `E1_A_UART_TX.ino`:

```

1 void setup() {
2   delay(400); //<-- buena práctica
3   Serial.begin(115200); // Inicializa la comunicación serial
4   Serial.println("\n"); // Envía un salto de línea (\n)
5   Serial.println("Hola Equipo IoT !!");
6 }
7
8 char letra = 'a';
9 void loop() {
10  Serial.println(letra++); // Envía letra y la incrementa en
11  delay(1000);             // Espera 1 segundo antes de enviar
12 }
13

```

Below the code editor, the **Monitor Serie** window is open. It shows the output of the serial communication:

```

Hola Equipo IoT !!
a
b
c
d

```

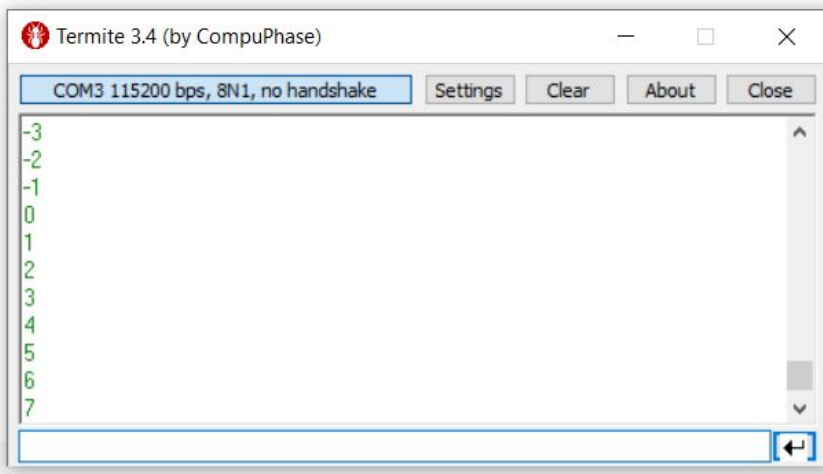
The **Monitor Serie** window also includes a text input field for messages, a dropdown for line endings (set to "Nueva línea"), and a dropdown for the baud rate (set to "115200 baud").

## *Ejemplo Transmisión, ciclo FOR*

```
void setup() {  
    delay(400); //<-- buena práctica  
    Serial.begin(115200); // Inicializa la comunicación serial  
    Serial.println("\nHola Equipo IoT !!");  
}  
  
void loop() {  
    for(int i=0; i<=100; i++){  
        Serial.println(i);  
        delay(50);  
    }  
    delay(2000);  
    for(int i=100; i>=0; i--){  
        Serial.println(i);  
        delay(50);  
    }  
    delay(2000);  
}
```

## Ejemplo Transmisión, ciclo WHILE

```
#define SW1 D0
#define SW2 D3
void setup() {
    delay(400); //<-- buena práctica
    Serial.begin(115200); // Inicializa la comunicación serial a 115200 baudios
    Serial.println("\nHola Equipo IoT !!");
}
int i = 0;
void loop() {
    while(digitalRead(SW1)==LOW){
        Serial.println(i++);
        delay(50);
    }
    while(digitalRead(SW2)==LOW){
        Serial.println(i--);
        delay(50);
    }
}
```



## Ejemplo Recepción

E1\_B\_\_UART\_TX\_RX.ino

```

1  void setup() {
2      delay(400);
3      Serial.begin(115200); // Inicializa la comunicación serial a 115200 baudios
4      Serial.println("\n"); // Envía un salto de línea (\n)
5      Serial.println("Hola Equipo EducaTR3BOL!!");
6      Serial.println("Prueba de RECEPCION de comandos");
7      pinMode(LED_BUILTIN,OUTPUT);
8  }
9
10 void loop() {
11     if(Serial.available() > 0 )
12     {
13         char letra = Serial.read();
14         Serial.print("Recibo: ");
15         Serial.println(letra);
16         if(letra=='A')    digitalWrite(LED_BUILTIN,LOW);
17         else if(letra=='B') digitalWrite(LED_BUILTIN,HIGH);
18     }
19 }
```

## Ejemplo conexión WiFi

```
#include <ESP8266WiFi.h>
const char* ssid    = "TR3BOL";    //Red WiFi
const char* password = "12345678"; //Contraseña Red WiFi
void setup() {
    delay(400);
    Serial.begin(115200);
    Serial.println("\n");
    WiFi.begin(ssid, password);
    delay(10);
    Serial.print("Intentando conectar a red ");
    Serial.println(ssid);
    while (WiFi.status() != WL_CONNECTED) { //Mientras no esté conectado
        Serial.print(".");
        delay(500);
    }
    Serial.println("");
    Serial.println("WiFi CONECTADO!!!  :)");
    Serial.print("IP address: ");
    Serial.println(WiFi.localIP());
}
void loop() {
    delay(2000);
    Serial.println("\nLISTO PARA SUBIR A LA NUBE!!");
}
```



## *PWM con funciones de librería*

```
#define LED_1 D4
#define LED_2 D8

void setup() {
    pinMode(LED_1,OUTPUT);
    pinMode(LED_2,OUTPUT);
}

void loop() {
    analogWrite(LED_1,75);
    analogWrite(LED_2,75);
    delay(2000);
    analogWrite(LED_1,125);
    analogWrite(LED_2,125);
    delay(2000);
    analogWrite(LED_1,255);
    analogWrite(LED_2,255);
    delay(2000);
}
```



## Ejemplo Integrador

```
#define analogPIN    A0
#define LED_1        D4
#define LED_2        D8
#define PULSADOR_1   D0
#define PULSADOR_2   D3

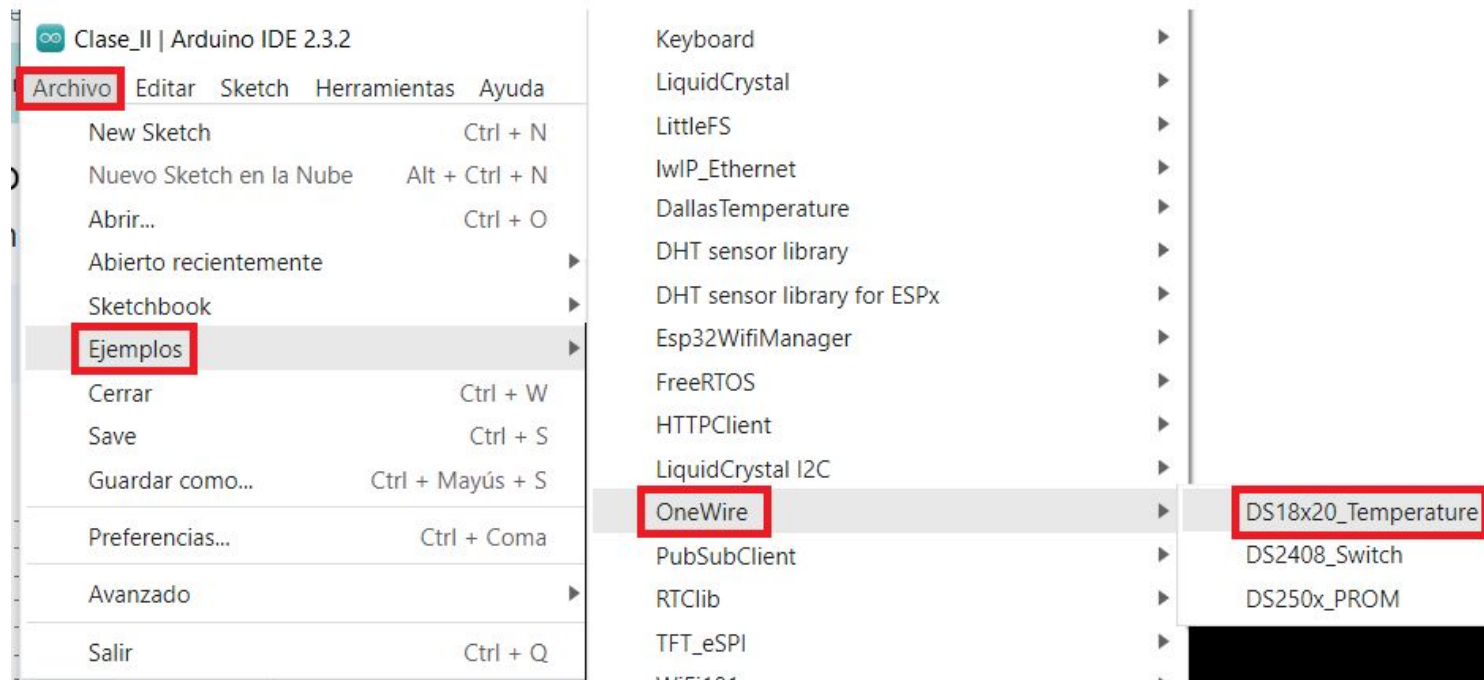
void setup() {
    delay(400);
    pinMode(LED_1,OUTPUT);
    pinMode(LED_2,OUTPUT);
    Serial.begin(115200);
    Serial.println("\nPrueba Analog, PWM e Input");
}

int analog = 0;
int PWM = 0;
```

```
void loop() {
    // ----- Analógico ----- //
    analog = analogRead(analogPIN);
    Serial.println(analog);
    // ----- PWM ----- //
    if(analog<20) analog = 0;
    PWM = 255*analog/1024.0;
    analogWrite(LED_1, PWM);
    analogWrite(LED_2, PWM);
    // ----- inputs ----- //
    if(digitalRead(PULSADOR_1)==LOW) Serial.println("SW1!!");
    if(digitalRead(PULSADOR_2)==LOW) Serial.println("SW2!!");
    delay(250);
}
```

## SONDA T° DS18B20

Rango de temperatura: -55 a 125°C



## SONDA T° DS18B20

```
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS D6 // Pin donde está conectado el sensor DS18B20
OneWire oneWire(ONE_WIRE_BUS); // Inicializa instancia de clase OneWire para comunicarse con el sensor
DallasTemperature sensors(&oneWire); // Ídem con DallasTemperature para interactuar con el sensor
void setup() {
    delay(400);
    Serial.begin(115200);
    Serial.println("Iniciando DS18B20...");
    sensors.begin(); // Inicializa la comunicación con los sensores DS18B20
}
void loop() {
    sensors.requestTemperatures(); // Solicita lectura de T° a todos los sensores conectados
    float temperatureCelsius = sensors.getTempCByIndex(0); // Lee T° del primer sensor encontrado
    if (temperatureCelsius == -127.00){ // Verifica si la lectura fue exitosa
        Serial.println("Error al leer la temperatura.");
    }
    else {
        Serial.print("DS18B20 OK\t\tTemp = ");
        Serial.print(temperatureCelsius);
        Serial.println(" C");
    }
    delay(2000); // Espera 2 segundos antes de la próxima lectura
}
```

## SONDA T° DS18B20

```
#include "miSondaTemperatura.h"

void setup() {
    delay(400);
    Serial.begin(115200);
    configuraSondaTemperatura();
}

void loop() {
    float temperatura = leeSondaTemperatura();
    Serial.print("Temperatura = ");
    Serial.print(temperatura);
    Serial.println("  C");
    delay(2000);
}
```

miSondaTemperatura.h

```
#include <OneWire.h>
#include <DallasTemperature.h>
#define ONE_WIRE_BUS D6 // Pin donde está conectado el sensor
OneWire oneWire(ONE_WIRE_BUS); // Inicializa instancia de
DallasTemperature sensors(&oneWire); // Ídem con DallasT

void configuraSondaTemperatura()
{
    Serial.println("Iniciando DS18B20...");
    sensors.begin(); // Inicializa la comunicación con los

float leeSondaTemperatura()
{
    sensors.requestTemperatures(); // Solicita lectura de
    float temperatureCelsius = sensors.getTempCByIndex(0);
    if (temperatureCelsius == -127.00){ // Verifica si la
        Serial.println("Error al leer la temperatura.");
    }
    return temperatureCelsius;
}
```