

Funciones esenciales Arduino

Funciones de Comunicación

- **Serial.begin(velocidad):** Inicializa la comunicación serial a una velocidad determinada (baudios). Se utiliza para enviar y recibir datos a través de un puerto serie.
Las velocidades que se suelen usar son 115200 o 9600 bps (bits por segundo).
- **Serial.print(valor):** Envía datos a través del puerto serie.
- **Serial.println(valor):** Envía datos a través del puerto serie y añade un salto de línea.
- **Serial.available():** Devuelve el número de bytes disponibles para leer en el búfer de recepción serial (avanzado, no se suele usar).

Ejemplo:

```
void setup() {  
    Serial.begin(9600);           // Configura el serial para 9600 bps  
}  
void loop() {  
  
    Serial.println("Hola desde Arduino!"); // Escribe en el serial  
}
```

Funciones de Entrada/Salida (I/O)

- **pinMode(pin, modo)**

Establece el modo de operación de un pin digital. Se configura como:

INPUT: El pin marcado se convierte en una entrada, pudiendo leer el valor que está siendo aplicado a ese pin (por ejemplo, si un botón está presionado o no).

OUTPUT: El pin se convierte en una salida, pudiendo aplicar voltaje en el pin (5V o 0V), como un LED, un motor, un relé...

Ejemplo:

```
pinMode(13, OUTPUT);    // Configuramos el pin 13 como salida  
pinMode(A1, INPUT);     // Configuramos el pin A1 como entrada
```

- **digitalWrite(pin, estado)**

Escribe en un pin digital configurado como salida. Puedes establecer el estado del pin en HIGH (alto, 5V) o LOW (bajo, 0V) para encender o apagar un LED, activar un relé, etc.

Ejemplo:

```
pinMode(13, OUTPUT);    // Configuramos el pin 13 como salida  
digitalWrite(13, HIGH); // Activamos el pin 13 con 5V
```

- **digitalRead(pin)**

Lee el estado de un pin digital como entrada. Devuelve HIGH si el pin está conectado a 5V y LOW si está conectado a tierra. Se utiliza para leer el estado de un botón, sensor de movimiento, etc.

Ejemplo:

```
bool valor = 0; // Inicialización de variable
pinMode(10, INPUT); // Configuramos el pin 10 como entrada
valor = digitalRead(10); // Asignamos a la variable el valor del pin 10
```

- **analogRead(pin)**

Lee el valor analógico de un pin. Los valores analógicos escalan entre 0 y 1023, y se utilizan para leer sensores analógicos como potenciómetros, sensores de luz, etc. Los pines para leer valores analógicos son los A0, A1, A2, A3.

Ejemplo:

```
int valor = 0; // Inicialización de variable
pinMode(A1, INPUT); // Configuramos el pin A1 como entrada
valor = analogRead(A1); // La variable obtiene el valor analógico del pin
```

- **analogWrite(pin, valor)**

Escribe un valor analógico en un pin PWM (modulación por ancho de pulso, siendo estos A0, A1, A2, A3). Se utiliza para controlar la intensidad de un LED, la velocidad de un motor, etc. Aquí, los valores van desde 0 hasta 255.

Ejemplo:

```
int valor = 453; // Inicialización de variable
pinMode(A1, OUTPUT); // Configuramos el pin A1 como salida
valor = analogWrite(A1, valor); // Asignamos al pin el valor de la variable
```

Funciones de Control de Tiempo

- **delay(milisegundos):** Pausa la ejecución del programa durante el número especificado de milisegundos.
- **millis():** Devuelve el número de milisegundos transcurridos desde que se inició el programa. Se utiliza para crear temporizadores o medir intervalos de tiempo.

Ejemplo:

```
digitalWrite(13, HIGH); // 5V al pin 13
delay(1000); // Espera 1 segundo
digitalWrite(13, LOW); // 0V al pin 13
delay(1000); // Espera 1 segundo
unsigned long currentMillis = millis(); // Tendrá un valor >= 2000 milisegundos
```

Funciones de Control de Flujo

- **if, else:** Permiten ejecutar un bloque de código según una o varias condiciones.

Ejemplo:

```
void loop() {  
    int sensorValue = analogRead(A0); // Lee valor del sensor en A0  
  
    if (sensorValue > 512) {           // Si el valor es >512  
        Serial.println("El valor es >512");  
    } else {                           //Si no,  
        Serial.println("El valor es <512");  
    }  
}
```

- **for:** Permite repetir un bloque de código un número determinado de veces.

Ejemplo:

```
void loop() {  
    for (int i = 0; i < 8; i++) {  
        Serial.println("Esta es la iteración " + String(i));  
    }  
}
```

Imprimiría por el serial:

"Esta es la iteración 0"

"Esta es la iteración 1"

...

"Esta es la iteración 7"

- **while:** Permite repetir un bloque de código mientras una o varias condiciones sean verdadera.

Ejemplo:

```
void loop() {  
    while (digitalRead(10) == LOW) {  
        Serial.println("Estoy esperando a que el botón del pin 10 se pulse");  
        delay(100);  
    }  
    Serial.println("El botón del pin 10 se ha pulsado");  
}
```

Funciones matemáticas

- `map(valor, limite_inferior_rango_X, limite_superior_rango_X, limite_inferior_rango_Y, limite_superior_rango_Y) :`
Transforma un valor de un rango X (del que partes) a un rango Y (al que quieres llegar), especificando los límites inferiores y superiores de ambos.

Ejemplo:

```
int valor_potenciometro = 0;
```

```
Servo miServo;
```

```
miServo.attach(3);
```

```
valor_potenciometro = analogRead(A0); // el valor valdrá desde 0 a 1023
```

```
miServo.write(valor_potenciometro) //SERÍA INCORRECTO, ya que no tiene 1023º
```

```
valor_potenciometro = map (valor_potenciometro, 0, 1023, 0, 180); // CORRECTO,
```

```
ahora el valor_potenciometro valdría entre 0 y 180 proporcionalmente al valor que tenía
```

