

# MicroPython sin IoT

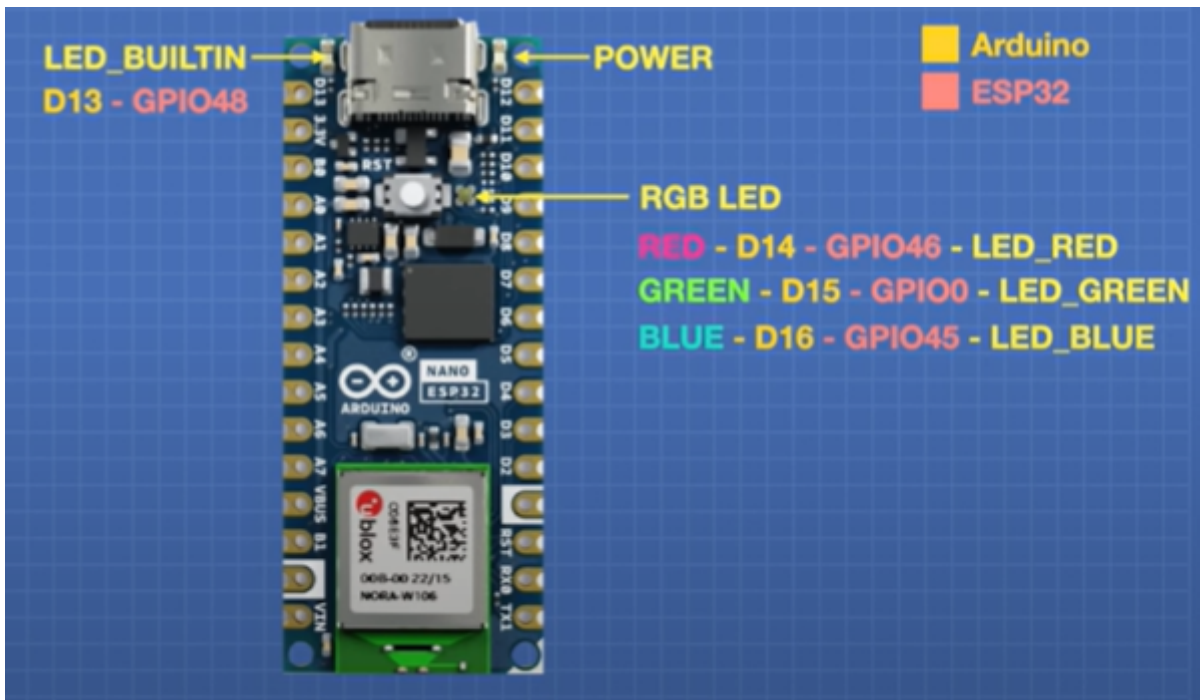
- [Parpadeo LED ESP32](#)
- [Parpadeo leds Alvik](#)
- [Danza](#)
- [Control con la mano](#)
- [Sigue líneas](#)
- [Evita obstáculos](#)
- [Programas de test](#)
- [Robótica para infantil](#)
- [Manejando servos](#)

# Parpadeo LED ESP32

## Objetivo

Vamos a hacer que parpadee el RGB integrado que tiene el ESP32 concretamente el color verde.

## Mapa de los pines en el Arduino Nano ESP32

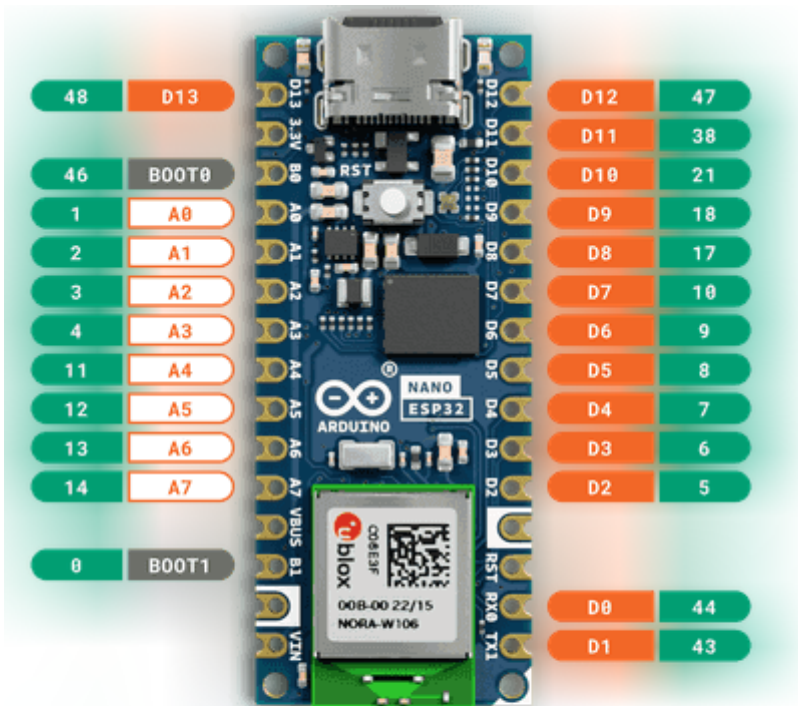


Extraído de [Youtube Exploring the Arduino Nano ESP32](#)

Como podemos observar, nuestro objetivo pues es el GPIO0

## ¿Dónde está físicamente los GPIO ?

Pues como podemos ver en este esquema el GPIO0 está en el pin BOOT1



**i** ESP32 pin numbers

**i** Nano pin numbers

Fuente <https://docs.arduino.cc/tutorials/alvik/user-manual/>

- SI USAMOS MICROPYTHON TENEMOS QUE USAR LAS VERDES
- SI USAMOS CÓDIGO ARDUINO IDE TENEMOS QUE USAR LAS ROJAS

Por eso, tal y como vimos en Poner la placa en modo Bootloader, el método alternativo enciende el led D1 de forma "hardware" al poner BOOT1 a GND para ponerlo en modo Bootloader, provoca el encendido del RGB en color verde.

## Programa

#extraído de <https://youtu.be/R51tf66es9w?t=1540>

```
from machine import Pin
import time

myLED = Pin(0,Pin.OUT)
```

```
while True:
    myLED.value(0)
    time.sleep(0.5)
    myLED.value(1)
    time.sleep(0.5)
```

## Aclaraciones

- Al hacer `from machine import Pin` estamos importando las definiciones input output de los pines del ESP32 nano arduino
- Ya hemos visto que lo que nos interesa es el 0 y lo ponemos como OUT

### ¿Y si queremos que parpadee el RGB en color ROJO qué cambiamos?

Easy peasy, cambiamos **`myLED = Pin (0, Pin.OUT)`** por **`myLED = Pin (46, Pin.OUT)`**

Que como puedes ver coincide también con un pin de poner en modo Bootloader: el BOOT0

Curiosidad: Por eso si se resetea Arduino Alvik (al encender, o al hacer dos clicks en el botón) se encienden y se apagan varias veces el led RGB en colores rojo y verde, pues se están activando los BOOTS

### ¿Y si queremos que parpadee el led color VERDE que hay al lado del USB (LED BUILTIN) ? ¿Qué cambiamos?

Easy peasy, cambiamos **`myLED = Pin (0, Pin.OUT)`** por **`myLED = Pin (48, Pin.OUT)`**

## Resultado:

<https://www.youtube.com/embed/XlbX6xPrqNE>

# Parpadeo leds Alvik

## El programa

```
from arduino_alvik import ArduinoAlvik
from time import sleep
import sys

alvik = ArduinoAlvik()
alvik.begin()
sleep(5)

while True:
    alvik.left_led.set_color(1, 0, 0)
    alvik.right_led.set_color(1, 0, 0)
    sleep(1)
    alvik.left_led.set_color(0, 0, 0)
    alvik.right_led.set_color(0, 0, 0)
    sleep(1)
```

Origen: <https://courses.arduino.cc/explore-robotics-micropython/lessons/getting-started/>

## Resultado

[https://www.youtube.com/embed/d\\_cLdqU8Koo](https://www.youtube.com/embed/d_cLdqU8Koo)

# Danza

## Programa

```
from arduino_alvik import ArduinoAlvik
from time import sleep
import sys

alvik = ArduinoAlvik()
alvik.begin()
sleep(5)

while True:
    #Drive forward
    alvik.set_wheels_speed(10,10)
    sleep(2)
    #Turn left
    alvik.set_wheels_speed(0,20)
    sleep(2)
    #Turn right
    alvik.set_wheels_speed(20,0)
    sleep(2)
    #Drive backwards
    alvik.set_wheels_speed(-10,-10)
    sleep(2)
```

De <https://courses.arduino.cc/explore-robotics-micropython/lessons/getting-started/>

## Vídeo

[https://www.youtube.com/embed/3uW\\_2uUMuTc](https://www.youtube.com/embed/3uW_2uUMuTc)

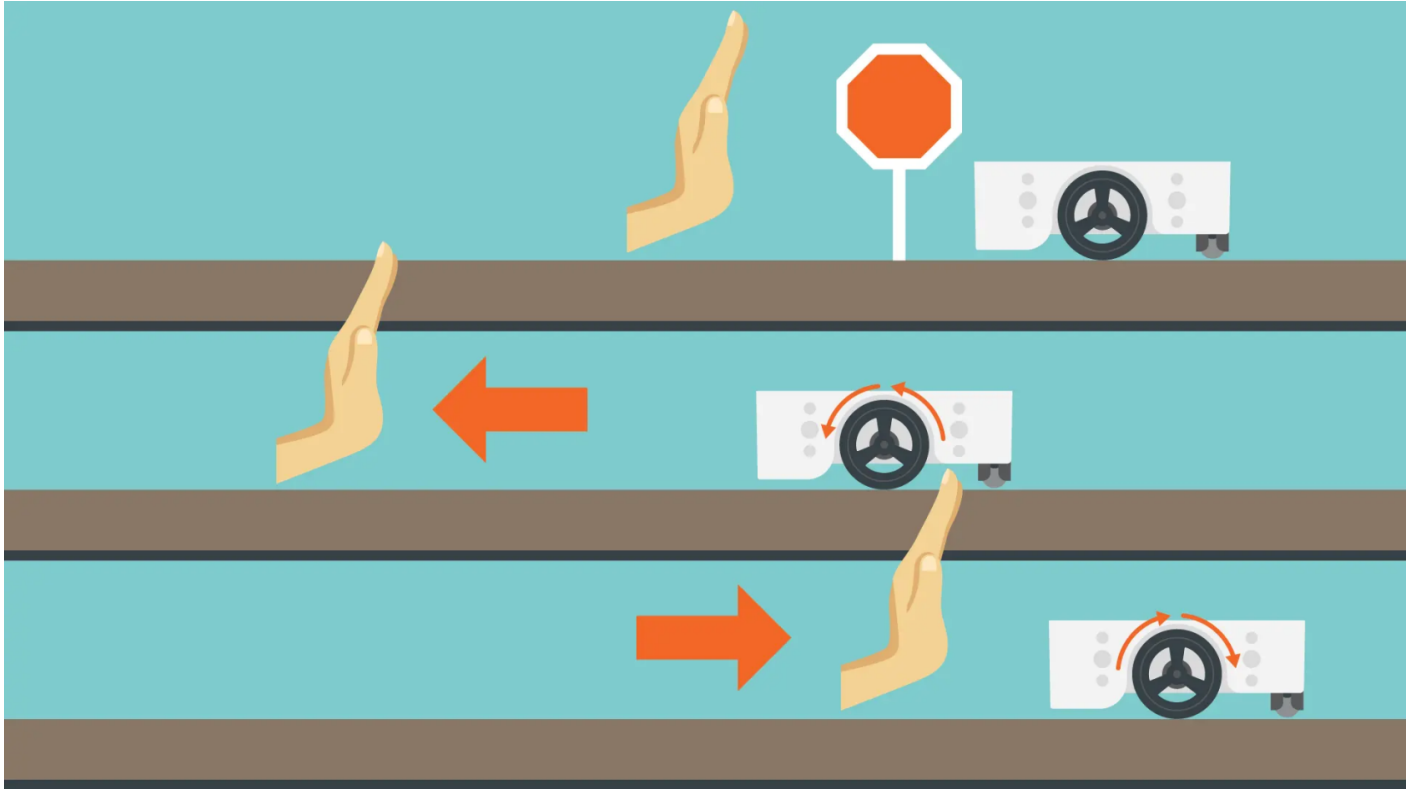
## Más caña

La instrucción `alvik.set_wheels_speed(0,20)` significa que da 0 rpm a la rueda izquierda y 20rpm a la derecha, donde rpm significa revoluciones por minuto ¿y si multiplicamos todos los rpm por 10?

<https://www.youtube.com/embed/rAAwrOV7CRo>

# Control con la mano

Aquí trataremos de hacer que ALVIK responda a la posición de nuestra mano a través de el sensor ultrasonidos que tiene en frente suyo. Tendrá que intentar permanecer en una distancia intermedia con lo que tiene enfrente suyo.



Fuente <https://docs.arduino.cc/tutorials/alvik/getting-started/> Author Jose Garcia

```
from arduino_alvik import ArduinoAlvik
from time import sleep
import sys

alvik = ArduinoAlvik()
alvik.begin()
sleep(5)

#ESTABLECER VELOCIDAD

speed = 30

#IMPRIMIR VALORES Y ESTABLECER VARIABLES
```



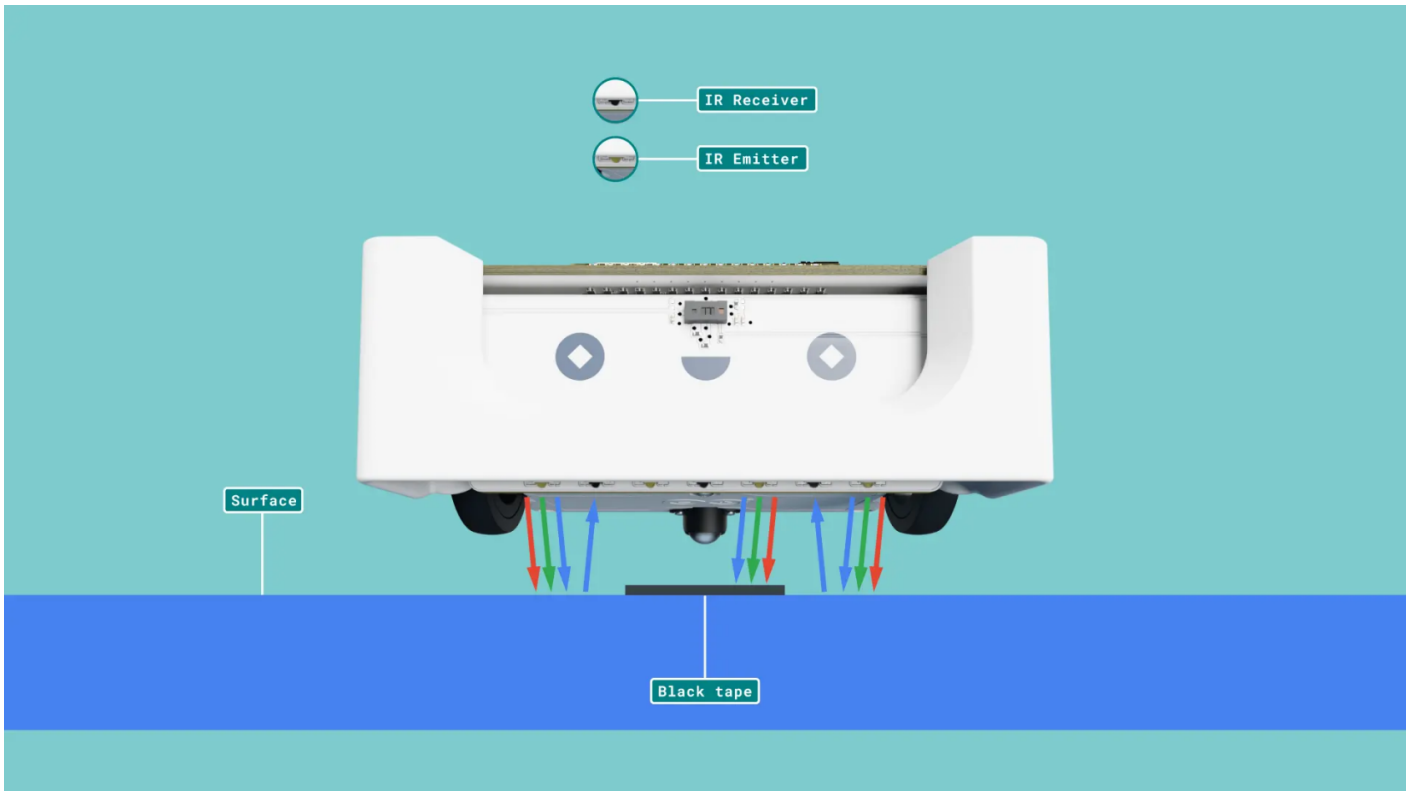
```
while True:
    try:
        center = alvik.get_distance_top()
        print(center)
        sleep(0.01)

        #Si la mano esta cerca, Alvik se va hacia atras
        if center <= 12:
            alvik.set_wheels_speed(-speed, -speed)
        #Si la mano esta lejos, Alvik se acerca
        elif center <= 30 and center >= 18:
            alvik.set_wheels_speed(speed, speed)
        #Si la mano esta en una distancia de 12-18, Alvik se queda quieto
        else:
            alvik.set_wheels_speed(0, 0)

    #INTERRUPCIÓN DEL USUARIO
    except KeyboardInterrupt as e:
        print('over')
        alvik.stop()
        sys.exit()
```

# Sigue líneas

Vamos a crear un programa SIGUE LÍNEAS. El objetivo es que el robot ALVIK sea capaz de seguir cualquier trazado de líneas utilizando sus sensores IR



Fuente <https://docs.arduino.cc/tutorials/alvik/getting-started/> Author Jose Garcia

Para poder conseguirlo simplemente tendremos que establecer distintas condiciones de que hacer dependiendo de que sensor del robot detecta el trazado negro.

En este programa también hemos hecho que el robot nos trasmita los valores de los distintos sensores y que el usuario pueda interrumpir el proceso (todo suponiendo que el robot está conectado al equipo).

```
from arduino_alvik import ArduinoAlvik
from time import sleep
import sys

alvik = ArduinoAlvik()
alvik.begin()
sleep(5)
```

```

#VELOCIDAD DEL ROBOT
base_speed = 30

#IMPRIMIR VALORES DE LOS SENSORES
while True:
    try:
        ir_left, ir_center, ir_right = alvik.get_line_sensors()
        print(ir_left, ir_center, ir_right)
        sleep(0.01)

        #Condiciones de giro, avance y parar
        if ir_center > 300:
            alvik.set_wheels_speed(base_speed, base_speed)
        elif ir_left > 300:
            alvik.set_wheels_speed(0, base_speed)
        elif ir_right > 300:
            alvik.set_wheels_speed(base_speed, 0)
        else:
            alvik.set_wheels_speed(0, 0)

    #INTERRUPCION DEL USUARIO
    except KeyboardInterrupt as e:
        print('over')
        alvik.stop()
        sys.exit()

```

No va muy preciso, el código es mejorable:

- Si la raya es fina, no avanza
- Si la raya no es negra no avanza
- La velocidad 30 es baja
- El margen límite 300 es demasiado generoso que hace que pueda quedarse quieto por detectar todo blanco (ver el final del vídeo)

<https://www.youtube.com/embed/6Nz4B13j3Vg>

# Evita obstáculos

## Programa

El núcleo del programa es la función api

```
get_distance(unit: str = 'cm')
```

Es sorprendente el sensor TOF como puede leer no sólo directamente sino a los lados :

- left\_tof: 45° to the left object distance
- center\_left\_tof: 22° to the left object distance
- center\_tof: center object distance
- center\_right\_tof: 22° to the right object distance
- right\_tof: 45° to the right object distance

El programa es extraído de <https://docs.arduino.cc/tutorials/alvik/getting-started/> Author Jose Garcia

```
from arduino_alvik import ArduinoAlvik
from time import sleep_ms
import sys

alvik = ArduinoAlvik()
alvik.begin()
sleep_ms(5000) # waiting for the robot to setup
distance = 20
degrees = 45.00
speed = 50.00

while (True):

    distance_l, distance_cl, distance_c, distance_r, distance_cr = alvik.get_distance()
    sleep_ms(50)
    print(distance_c)

    if distance_c < distance:
        alvik.rotate(degrees, 'deg')
    elif distance_cl < distance:
        alvik.rotate(degrees, 'deg')
```

```
elif distance_cr < distance:
    alvik.rotate(degrees, 'deg')
elif distance_l < distance:
    alvik.rotate(degrees, 'deg')
elif distance_r < distance:
    alvik.rotate(degrees, 'deg')
else:
    alvik.drive(speed, 0.0, linear_unit='cm/s')
```

## Resultado

El código es mejorable, pues que rote 45 grados tantas veces puede hacer que se quede "enganchado" en una esquina, ver el final del vídeo:

<https://www.youtube.com/embed/CWx501rFpyA>

# Programas de test

En el repositorio <https://github.com/arduino/arduino-alvik-mpy/tree/main/examples> podemos encontrar ejemplos para ver el uso de los diferentes sensores y actuadores, por ejemplo

Sensor name	Part name	Test program name
RGB Color detection	APDS 9660	read_color_sensor.py
ToF 8x8 Array - up to 350 cm	LSM6DSOX	read_tof.py
IMU - 6 degree	VL53L7CX	read_imu.py
3x Line follower	custom made	line_follower.py
7x Touch sensor	AT42QT2120	read_touch.py
Actuator name	Part name	Test program name
Geared motors w/ encoder	GM12-N20VA-08255-150-EN	wheels_positions.py
RGB LEDs	RGB LEDs	leds_settings.py

## Detector de color

Modificación del read\_color\_sensor.py

```
from arduino_alvik import ArduinoAlvik
from time import sleep_ms
import sys

alvik = ArduinoAlvik()
alvik.begin()

while True:
    try:
        r, g, b = alvik.get_color()
        h, s, v = alvik.get_color('hsv')
        print(f'RED: {r}, Green: {g}, Blue: {b}, HUE: {h}, SAT: {s}, VAL: {v}')
        print(f'COLOR LABEL:')
        print ({alvik.get_color_label()})
        sleep_ms(1000)
    except KeyboardInterrupt as e:
        print('over')
```

```
alvik.stop()
sys.exit()
```

<https://www.youtube.com/embed/j0tgpjJKK40>

## Detector TOF

Si ejecutamos read\_tof.py

```
from arduino_alvik import ArduinoAlvik
from time import sleep_ms
import sys

alvik = ArduinoAlvik()
alvik.begin()

while True:
    try:
        L, CL, C, CR, R = alvik.get_distance()
        T = alvik.get_distance_top()
        B = alvik.get_distance_bottom()
        print(f'T: {T} | B: {B} | L: {L} | CL: {CL} | C: {C} | CR: {CR} | R: {R}')
        sleep_ms(100)
    except KeyboardInterrupt as e:
        print('over')
        alvik.stop()
        sys.exit()
```

Detecta hasta los obstáculos por arriba

<https://www.youtube.com/embed/XHXCfbIL4ks>

## Giro

Si ejecutamos read\_imu.py

```
from arduino_alvik import ArduinoAlvik
from time import sleep_ms
```

```
import sys

alvik = ArduinoAlvik()
alvik.begin()

while True:
    try:
        ax, ay, az = alvik.get_accelerations()
        gx, gy, gz = alvik.get_gyros()
        print(f'ax: {ax}, ay: {ay}, az: {az}, gx: {gx}, gy: {gy}, gz: {gz}')
        sleep_ms(100)
    except KeyboardInterrupt as e:
        print('over')
        alvik.stop()
        sys.exit()
```

Vemos como el eje x cambia de -1 0 a 1 según la posición

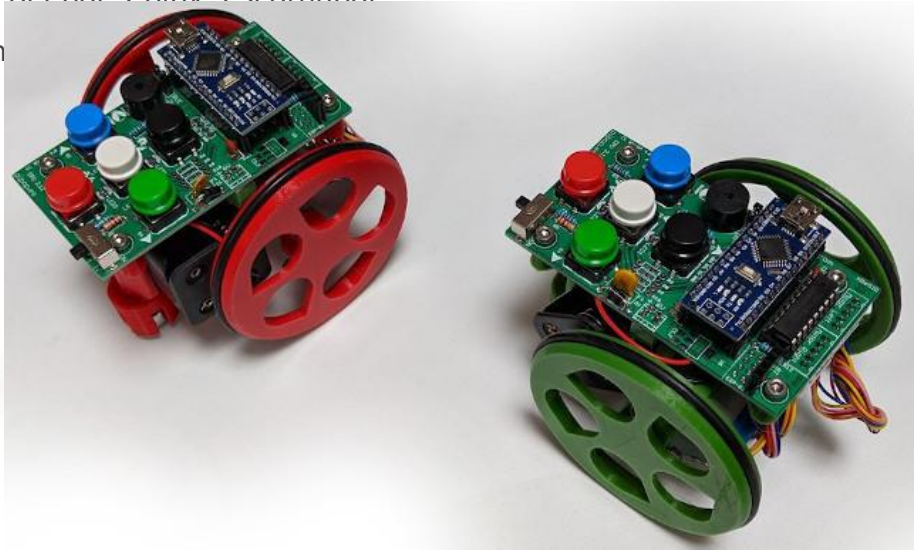
[https://www.youtube.com/embed/E083Xe\\_IMFY](https://www.youtube.com/embed/E083Xe_IMFY)



# Robótica para infantil

Se puede hacer un robot tipo Beebot, Colby, Escornabot

Si no conocéis estos robots m



Podemos cargar el siguiente programa, modificado de [https://github.com/arduino/arduino-alvik-mpy/blob/main/examples/touch\\_move.py](https://github.com/arduino/arduino-alvik-mpy/blob/main/examples/touch_move.py)

```
from arduino_alvik import ArduinoAlvik
from time import sleep_ms
import sys

alvik = ArduinoAlvik()
alvik.begin()

alvik.left_led.set_color(1, 0, 0)
alvik.right_led.set_color(1, 0, 0)

distancia = 15

movements = []

def blink():
    alvik.left_led.set_color(1, 0, 1)
```

```
alvik.right_led.set_color(1, 0, 1)
sleep_ms(200)
alvik.left_led.set_color(1, 0, 0)
alvik.right_led.set_color(1, 0, 0)
```

```
def add_movement():
    global movements

    if alvik.get_touch_up():
        movements.append('forward')
        blink()
        while alvik.get_touch_up():
            sleep_ms(100)
    if alvik.get_touch_down():
        movements.append('backward')
        blink()
        while alvik.get_touch_down():
            sleep_ms(100)
    if alvik.get_touch_left():
        movements.append('left')
        blink()
        while alvik.get_touch_left():
            sleep_ms(100)
    if alvik.get_touch_right():
        movements.append('right')
        blink()
        while alvik.get_touch_right():
            sleep_ms(100)
    if alvik.get_touch_cancel():
        movements = []
        for i in range(0, 3):
            val = i % 2
            alvik.left_led.set_color(val, 0, 0)
            alvik.right_led.set_color(val, 0, 0)
            sleep_ms(200)
        while alvik.get_touch_cancel():
            sleep_ms(100)
```

```

def run_movement(movement):
    if movement == 'forward':
        alvik.move(distancia, blocking=False)
    if movement == 'backward':
        alvik.move(-distancia, blocking=False)
    if movement == 'left':
        alvik.rotate(90, blocking=False)
    if movement == 'right':
        alvik.rotate(-90, blocking=False)
    while not alvik.get_touch_cancel() and not alvik.is_target_reached():
        alvik.left_led.set_color(1, 0, 0)
        alvik.right_led.set_color(1, 0, 0)
        sleep_ms(100)
        alvik.left_led.set_color(0, 0, 0)
        alvik.right_led.set_color(0, 0, 0)
        sleep_ms(100)

while alvik.get_touch_ok():
    sleep_ms(50)

while not (alvik.get_touch_ok() and len(movements) != 0):
    add_movement()
    sleep_ms(50)

try:
    while True:
        alvik.left_led.set_color(0, 0, 0)
        alvik.right_led.set_color(0, 0, 0)
        for move in movements:
            run_movement(move)
            if alvik.get_touch_cancel():
                break

    movements = []

    while not (alvik.get_touch_ok() and len(movements) != 0):
        alvik.left_led.set_color(1, 0, 0)
        alvik.right_led.set_color(1, 0, 0)
        alvik.brake()
        add_movement()

```

```
        sleep_ms(100)
except KeyboardInterrupt as e:
    print('over')
    alvik.stop()
    sys.exit()
```

El resultado es que perfectamente se puede usar como robótica en infantil

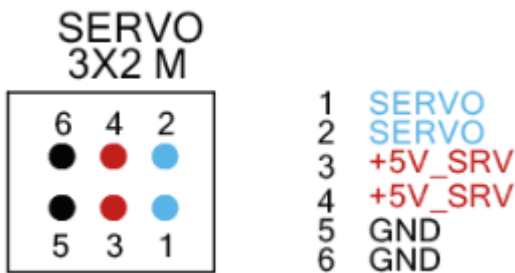
Los robots Beebot, Colby, Escornabot. utilizan la distancia de 15cm de desplazamiento, justo lo mismo que los palos depresores de lengua, luego fácilmente uno puede hacer un circuito :

<https://www.youtube.com/embed/h8wAwcPxYL0>

# Manejando servos

## Conexión

Se pueden conectar hasta dos servos, el A es el de arriba y el B es el de abajo



## La api `set_servo_positions`

Nos permite controlar estos dos servos indicando el primer argumento el ángulo (0-180) del A y en el segundo el del B **`set_servo_positions(a_position: int, b_position: int)`**

## Programa

Extraído de <https://docs.arduino.cc/tutorials/alvik/user-manual/#add-servo-motors>

```
from arduino_alvik import ArduinoAlvik

import time

alvik = ArduinoAlvik()

alvik.begin()

while True:
    alvik.set_servo_positions(0,0)
    time.sleep(2)
    alvik.set_servo_positions(90,180)
    time.sleep(2)
    alvik.set_servo_positions(180,90)
    time.sleep(2)
    alvik.set_servo_positions(90,0)
```

```
time.sleep(2)
```

## Resultado

<https://www.youtube.com/embed/hloayWNgGao>