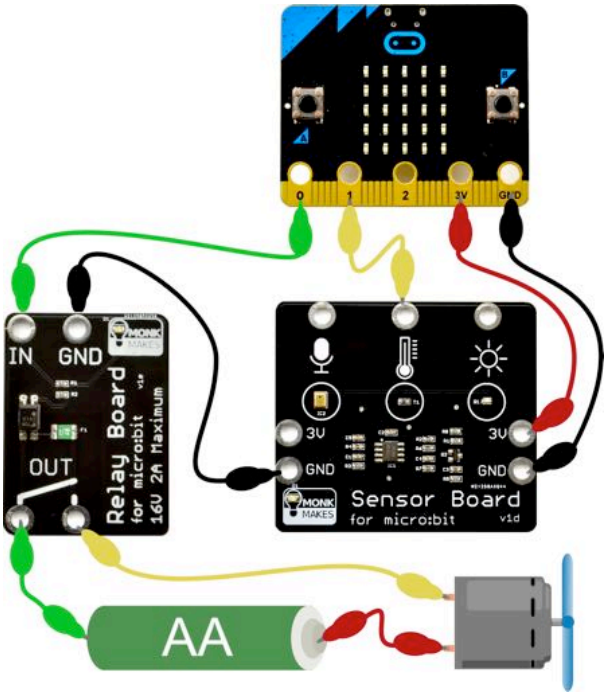
Instrucciones:

Kit de Iniciación a la Electrónica

Para micro:bit v2a.es



Rev 2a-es

Tabla de Contenidos

Componentes 3
Montando los Proyectos.....4
Empezando.....7
Proyecto 1. Alarma de Movimiento 11
Proyecto 2. Faro.....13
Proyecto 3. Gritómetro.....15
Proyecto 4. Termómetro.....18
Proyecto 5. Controlando la Velocidad de un Ventilador.....21
Proyecto 6. Ventilador Automático.....25
Proyecto 7. Música Mágica.....28
El Editor De Bloques de JavaScript 30
Usando MicroPython.....32
Resolución de problemas.....33
¿Qué Hacer para Seguir Avanzando?.....34
Notas.....35
MonkMakes.....36

Componentes

micro:bit, las pilas y el cable USB no se incluyen en el kit.
Antes de hacer nada más, comprueba que tu kit incluye todos estos componentes:

Cantidad		
1	Altavoz MonkMakes para micro:bit (reproduce sonidos de la micro:bit)	
1	Relé MonkMakes para micro:bit (enciende y apaga cosas o controla el brillo de una luz o la velocidad de un motor)	
1	Placa de Sensores MonkMakes para micro:bit (mide sonido, temperatura y luz)	
1	10 cables con pinzas cocodrilo	
1	Pequeño motor con ventilador	
1	Caja portapilas para pila AA (no se incluye la pila)	
1	Bombilla y portalámparas	

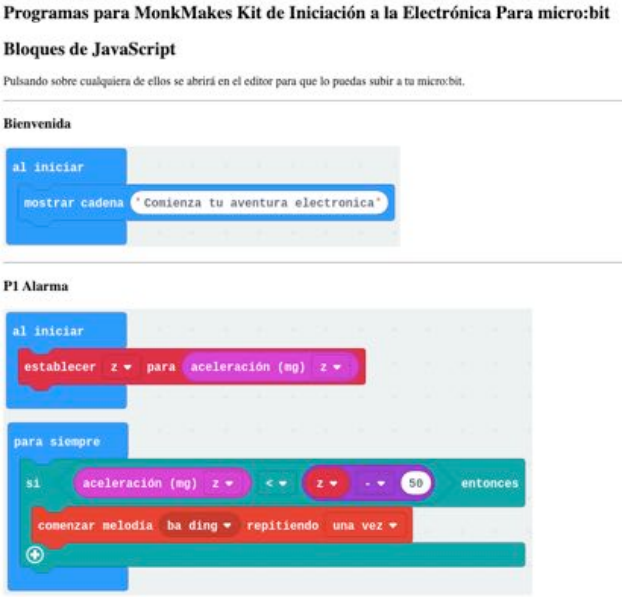
Montando los Proyectos

Programación

Antes de conectar las placas a tu micro:bit debemos subir el programa del proyecto. Con esto nos aseguramos que los pines están correctamente configurados como entradas o como salidas antes de conectar la electrónica.

Este kit viene con programas tanto para el editor de Bloques de JavaScript como para Micropython. Si la programación es algo nuevo para ti, tu mejor elección para empezar es el editor de Bloques. Si quieres usar la versión de Micropython de los programas, empieza primero por leer la sección Usando Micropython, más adelante en este libro.

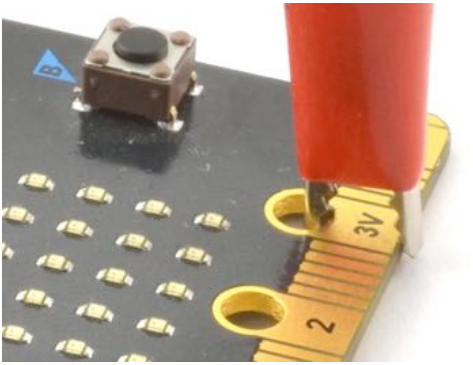
El editor de bloques te permite programar tu micro:bit directamente desde el navegador de tu ordenador. Puedes encontrar todos los programas en la página del software del kit: https://monkmakes.com/mb_es_2a.html como ves más abajo.



Pulsando sobre cualquiera de ellos se abrirá en el editor para que lo puedas subir a tu micro:bit.

Pinzas de cocodrilo

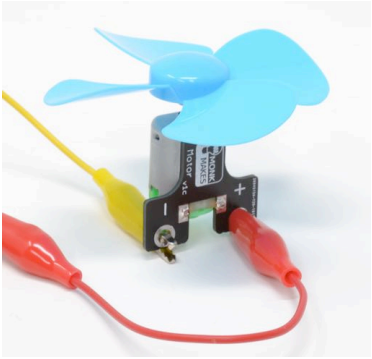
Montaremos los proyectos de este kit conectando tu micro:bit con una o más de las placas MonkMakes usando pinzas de cocodrilo. Tienes que ser cuidadoso al conectar las pinzas a la micro:bit. La mejor forma de conectarlas es verticalmente como se puede ver en la imagen de abajo.



Conectando las pinzas de esta manera evitas conexiones accidentales entre los conectores grandes con agujero y los otros más delgados (las líneas doradas de la imagen anterior)

El Motor

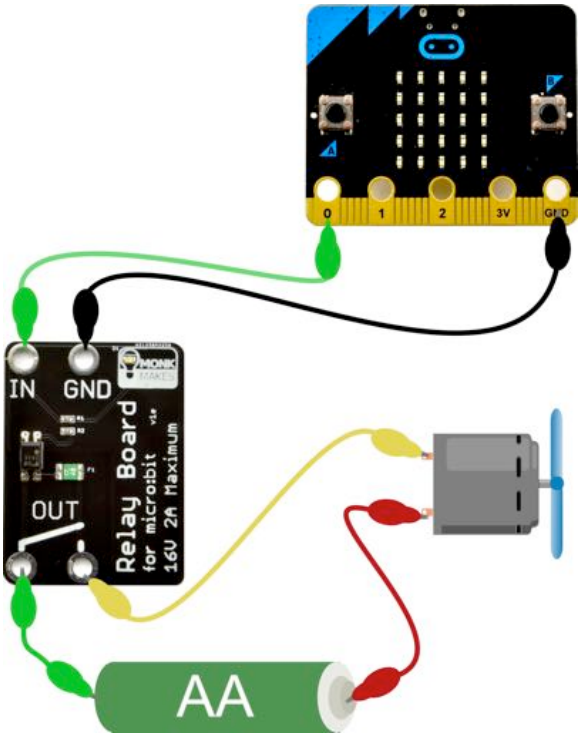
Para evitar que el motor se caiga, conéctelo usando los pinzas de cocodrilo como se muestra a continuación.



Construyendo un Proyecto

La micro:bit y las placas MonkMakes son bastante robustas frente a errores de conexión, pero siempre es buena idea desconectar tu micro:bit de tu ordenador mientras estás conectando todo. Cuando termines, revisa cuidadosamente todas las conexiones antes de conectar tu micro:bit a tu ordenador.

Cada proyecto incluye un esquema de conexión, como el que sigue en la siguiente página. En él se muestra qué componente se conecta a cuál. Se sugiere el color de los cables para cada conexión. El proyecto funcionará perfectamente, independientemente del color del cable usado, pero te será más sencillo entender cómo funciona si mantienes los mismos colores, especialmente el rojo para las conexiones de alimentación de 3V y el negro para las conexiones a tierra.



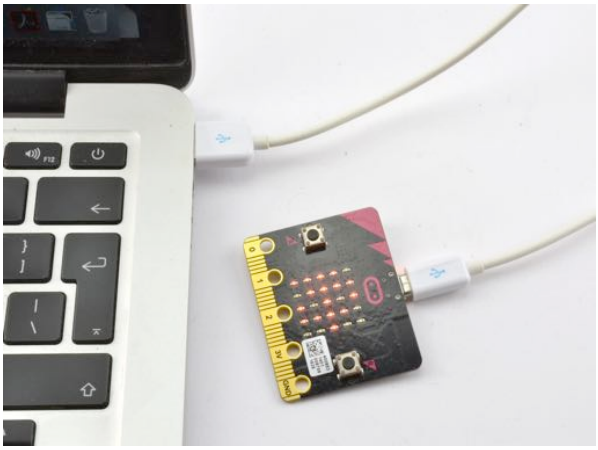
Empezando

Sigue los pasos que te indicamos a continuación para empezar con tu kit. No se usan ninguna de las placas MonkMakes, por lo que no vas a necesitar cables con pinzas de cocodrilos.

Probablemente puedes saltarte esta sección si ya has usado anteriormente la micro:bit.

Paso 1. Conecta tu micro:bit a tu ordenador.

Conecta tu micro:bit a un puerto USB libre de tu ordenador.



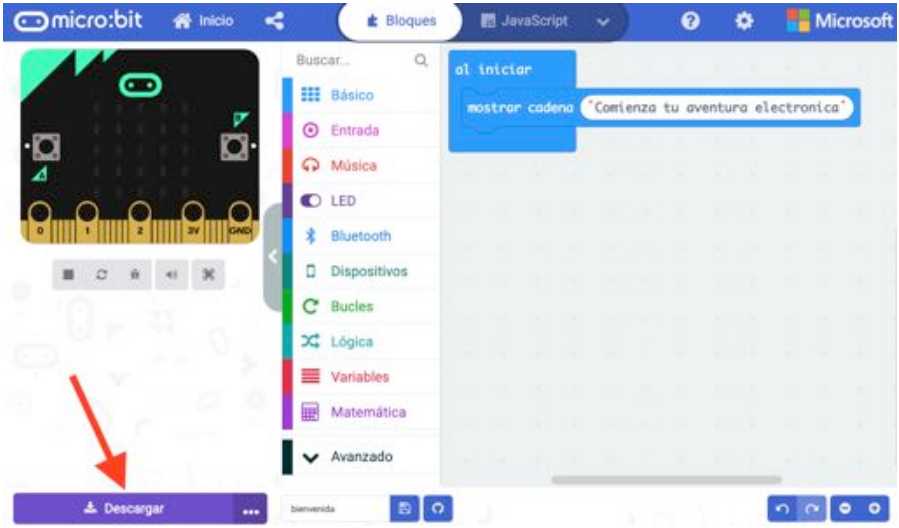
Si tu micro:bit es nueva, aparecerán mensajes mostrando las característica de micro:bit en la pantalla LED. Vamos a reemplazar el programa que se está ejecutando ahora mismo con uno nuevo.

Paso 2. Abre el editor de bloques.

Para subir un nuevo programa (que casualmente se llama *Bienvenida)* a tu micro:bit, primero tenemos que abrir el Editor de bloques. Lo puedes hacer entrando en la siguiente dirección de internet:
https://www.monkmakes.com/mb_es_1d.html

Y haciendo clic en el proyecto *Bienvenida* verás lo siguiente.

Hacemos clic en el enlace “Edit Code” y veremos lo siguiente



Paso 3. Instalar el Programa

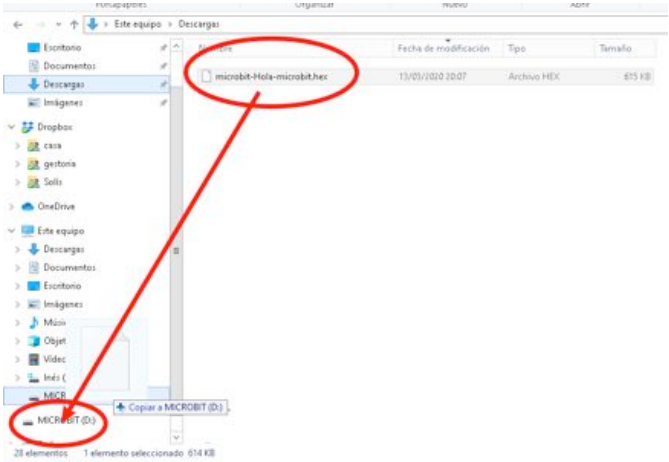
A continuación vamos a enviar el programa *Bienvenida* a la micro:bit.

En las nuevas versiones de micro:bit, al hacer clic en el enlace Descarga el programa se instala automáticamente en tu micro:bit. De lo contrario, habrá que instalarlo manualmente de esta manera:

Hacemos clic en el enlace “Descargar” en la parte de abajo de la página lo que

descargará un fichero con extensión *.hex* que contiene el programa. Aparecerá en el navegador la siguiente indicación.

Como nos indica la ventana, ahora tenemos que mover el fichero *hex* dentro de la micro:bit. Al conectarla, la micro:bit aparece en el ordenador como un disco USB. Es decir, tenemos que arrastrar el fichero hex desde la carpeta Descargas a la unidad MicroBit en el explorador de ficheros como si la micro:bit fuera un pen drive.



Si usas Mac, Linux o Raspberry Pi en tu ordenador el proceso es el mismo, pero tendrás que usar otra aplicación para copiar ficheros

Tan pronto como sueltes el fichero hex en el icono de la micro:bit, un LED amarillo, que está e la parte de atrás de la micro:bit empezará a parpadear. Decimos que hemos *reprogramado* la micro:bit. En cuanto que el led deje de parpadear, el programa *Bienvenida* comenzará a mostrar en la pantalla LED el siguiente mensaje: *Comienza tu aventura electrónica.*

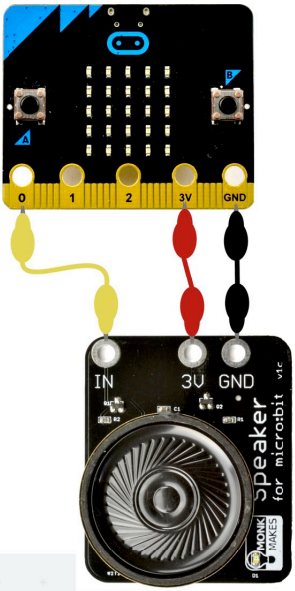
¡Enhorabuena! ¡Estás preparado para comenzar con el Proyecto 1!

Para aprender más sobre el Editor de bloques, más adelante tienes la sección *El Editor de Bloques*.

Proyecto 1. Alarma de Movimiento

Lo que necesitarás:

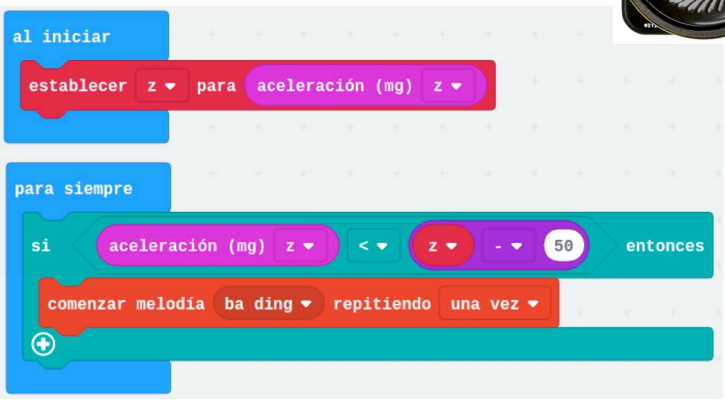
- micro:bit
- Altavoz MonkMakes para micro:bit
- Programa: *P1 Alarma*



Sube el programa *P1 Alarma* a tu micro:bit (en la página 8 tienes instrucciones sobre cómo subir tus programas a tu micro:bit). Después conecta el Altavoz como se muestra a la derecha.

Mueve tu micro:bit. Al hacerlo, deberías escuchar un sonido que indica que se ha disparado la alarma.

Cómo Funciona



El programa usa el acelerómetro que tiene incorporado la micro:bit para medir cualquier fuerza que actúe en la dirección z (verticalmente). Cuando la micro:bit se encuentra completamente plana deberíamos obtener el valor -1023, pero cuando la cogemos y la levantamos la fuerza cambiará.

Cuando el programa arranca, guarda el valor de la aceleración en una variable llamada z. En el bucle *para siempre* comprobamos la aceleración y si ésta ha bajado en más de 50 desde el valor inicial (z), la alarma suena.

Modificando el número 50, podrás ajustar la sensibilidad de la alarma. También puedes seleccionar diferentes músicas en el bloque '*comenzar melodía*'.

MicroPython Código

Si quieres usar la versión MicroPython de los programas en lugar de los bloques en el Editor de Bloques, revisa la sección casi al final del libro llamada *Usando MicroPython* para instrucciones sobre cómo descargar y usar el código.

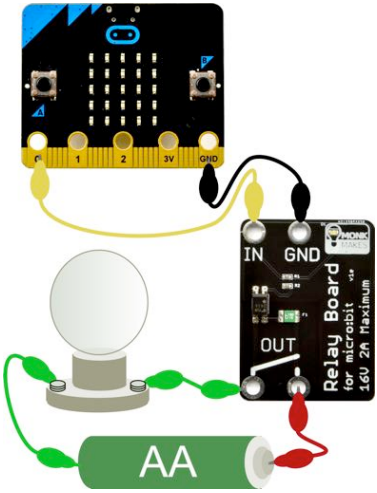
Este sería el código MicroPython para este proyecto.

```
from microbit import *
import music


z = accelerometer.get_z()


while True:
    if accelerometer.get_z() < z - 50:
        music.play(music.BA_DING)
```

Proyecto 2. Faro

Lo que necesitarás:

- micro:bit
- Relé MonkMakes para micro:bit
- Bombilla pequeña y portalámparas
- Pila AA y su portapilas
- Programa: *P2 Faro*



Sube el programa *P2 Faro* a tu micro:bit (ver página 8) y después conecta el Relé, la pila y la bombilla como puedes ver a la derecha.

El proyecto hará que la bombilla parpadee encendiéndose y apagándose.

Puedes hacer una estructura de cartón similar a un faro y fijar la bombilla en la parte superior.

Cómo Funciona

Este es el código del proyecto

Primero el Pin P0 se activa usando el bloque de ‘escritura digital’. El bloque 'pausa' asegura que se mantenga en este estado 1000 milisegundos (1 segundo) antes de que lo apaguemos. Tras una pausa de 2 segundos el ciclo se repite.

Código MicroPython

Si quieres usar la versión MicroPython de los programas en lugar de los bloques en el Editor de Bloques, revisa la sección casi al final del libro llamada *Usando MicroPython* para instrucciones sobre cómo descargar y usar el código.

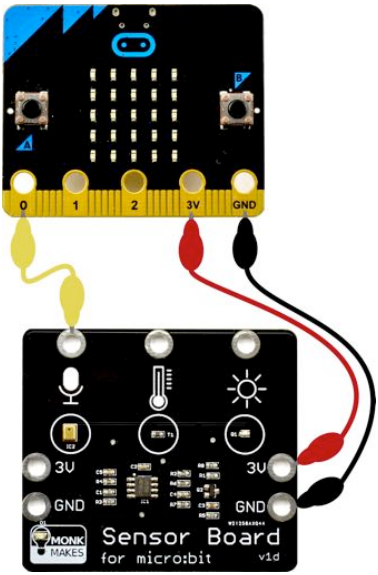
Este sería el código MicroPython para este proyecto.

```
from microbit import *


while True:
    pin0.write_digital(1)
    sleep(1000)
    pin0.write_digital(0)
    sleep(2000)
```

Proyecto 3. Gritómetro

Lo que necesitarás:

- micro:bit
- Placa de sensores MonkMakes para micro:bit
- Programa: *P3 Gritómetro*

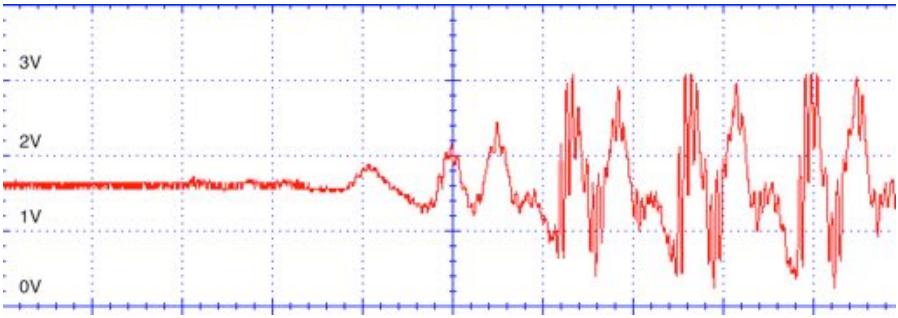


Sube el programa *P3 Gritómetro* a tu micro:bit (puedes ver cómo hacerlo en la página 8) y después conecta la placa de Sensores MonkMakes como ves a la derecha.

Haz ruido cerca de la parte del micrófono de la Placa de Sensores MonkMakes y podrás ver que los LEDs de la micro:bit se encienden mostrando el volumen. Cuando más fuerte sea el ruido más LEDs se encenderán.

Cómo funciona

El micrófono convierte el sonido en una señal eléctrica parecida a esta:

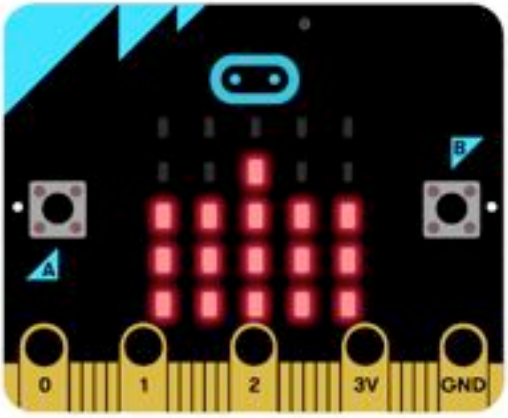


El voltaje en la salida del micrófono de la placa de sensores siempre estará entre 0 y 3V, la señal oscilará arriba y abajo del punto medio de 1,5V aproximadamente. El número de LEDs encendidos se calcula en función de cuánto se separa el voltaje del valor medio (1,5V).

Este es el código del proyecto:

La función *lectura analógica del pin* devuelve un número entre 0 y 1023 dependiendo del voltaje al que está sometido el pin. Puesto que el pin P0 está conectado a el micrófono, la señal variará entre 0 y 1023 cuando la amplitud de la onda de sonido varíe. Restamos 511 del valor leído para obtener el punto medio de la señal. Esto quiere decir que a veces obtendremos un barra de gráfico negativa, pero esto no importa porque el bloque que hace la gráfica ignora estos valores.

El bloque *plot bar graph* enciende más LEDs cuanto mayor es el número que se le proporciona. La imagen de abajo muestra el resultado para un determinado nivel de ruido



Código MicroPython

Si quieres usar la versión MicroPython de los programas en lugar de los bloques en el Editor de Bloques, revisa la sección casi al final del libro llamada *Usando MicroPython* para instrucciones sobre cómo descargar y usar el código.

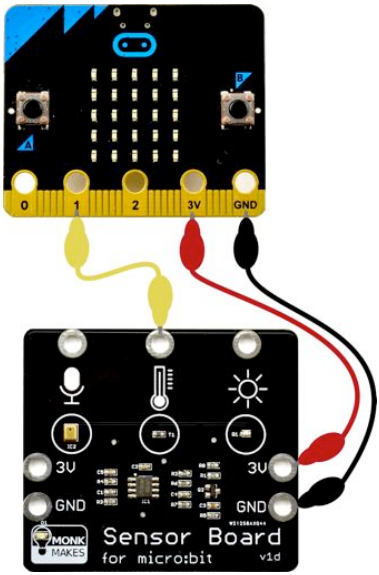
Este sería el código MicroPython para este proyecto.

```
from microbit import *


def graficobarras(a):
    display.clear()
    for y in range(0, 5):
        if a > y:
            for x in range(0, 5):
                display.set_pixel(x, 4-y, 9)


while True:
    nivel_sonido = (pin0.read_analog() - 511) / 100
    graficobarras(nivel_sonido)
```

Proyecto 4. Termómetro

Lo que necesitarás:

- micro:bit
- Placa de Sensores MonkMakes para micro:bit
- Programa: *P4 Termómetro* o *P4 Termómetro F* (para grados Fahrenheit)



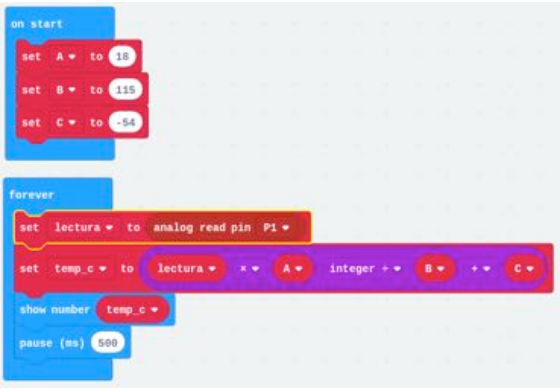
Sube el programa *P4 Termómetro (para grados C)* o *P4 Termómetro F* (para Fahrenheit) *a la micro:bit* (puedes ver en la página 8 cómo hacerlo). Después conecta la Placa de Sensores como se muestra.

Tu micro:bit mostrará la temperatura del sensor de la Placa de Sensores MonkMakes (indicado por el símbolo del termómetro).

Si pones tu dedo sobre el sensor verás como la temperatura comienza a subir.

Cómo funciona

El sensor de temperatura usa un componente llamado *termistor*. Un termistor es un componente cuya resistencia al paso de la corriente eléctrica varía en función de su propia temperatura.



Para este tipo de termistor, cuando mayor es la temperatura, menor es la resistencia al paso de la corriente. Este efecto es el que usamos para medir la temperatura.

Se necesitan cálculos matemáticos para convertir la lectura del termistor en grados C. Usa el programa *P4 Termómetro F* si prefieres ver el resultado en grados Fahrenheit.

Calibración

Esta sección te mostrará como puedes conseguir una medida más precisa de tu termómetro calibrándolo.

Para calcular la temperatura ya sea en grados F o C, primeramente multiplicamos la lectura analógica es por un número (A), el resultado se dividido por otro número (B) y después sumamos al resultado un número positivo o negativo (C). Para hacer más preciso tu termómetro puedes usar el programa *P4 Calibración* que te permitirá encontrar las lecturas analógicas para dos temperaturas conocidas (usando un segundo termómetro) y a partir de ellas calcular los valores de A, B y C.

Paso 1. Sube el programa *P4 Calibración* a tu micro:bit con la Placa de Sensores conectada como hemos visto más arriba.

Paso 2. Mide la temperatura (ya sea en F o C) usando un segundo termómetro colocado cerca del sensor de temperatura de la placa. Pulsa el botón A y anota el valor visualizado. Puedes escribir estos valores en la tabla de abajo, donde puedes ver los valores encontrados para la temperatura 18 grados y una lectura del sensor de 460.

Paso 3. Guarda un cubito de hielo en una bolsa de plástico y mantenlo a la vez junto al sensor de temperatura de la Placa de Sensores y al segundo termómetro. Espera unos 30 segundos y a continuación anota tanto la lectura del termómetro como la lectura que se muestra en tu placa micro:bit al pulsar el botón A. En nuestro caso las lecturas fueron 0 grados centígrados y 345. Anótalas en la tabla.

Paso 4. Calcula A, B y C usando las fórmulas que se muestran en la tabla

Temperatura ambiente usando el termómetro (t1)	18
Lectura del sensor a temperatura ambiente (r1)	460
Temperatura baja usando el termómetro (t2)	0
Lectura del sensor a temperatura baja (r2)	345
A = t1 – t2	18
B = r1 – r2	115
C = t2 – (A / B) * r2	0 – (18/115) * 345 = -54

Una vez calculados los valores de A B y C cambia estos valores en tu programa de bloques

Código MicroPython

Si quieres usar la versión MicroPython de los programas en lugar de los bloques en el Editor de Bloques, revisa la sección casi al final del libro llamada *Usando MicroPython* para instrucciones sobre cómo descargar y usar el código.

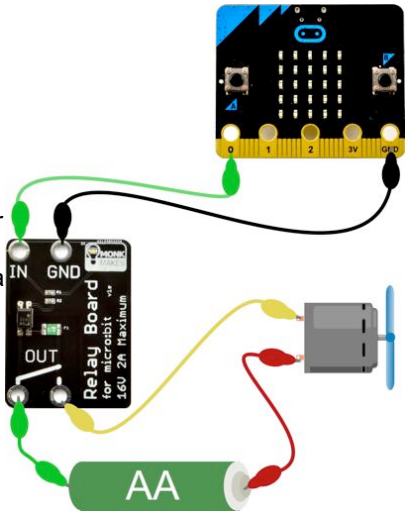
Este sería el código MicroPython para este proyecto.

```
from microbit import *


while True:
    lectura = pin1.read_analog()
    temperatura_c = round(lectura * 0.157 - 54)
    display.scroll(str(temperatura_c))
```

Proyecto 5. Controlando la Velocidad de un Ventilador

Lo que necesitarás:

- micro:bit
- Relé MonkMakes para micro:bit
- Pila AA y portapilas
- Motor con la hélice puesta
- Programa: *P5 Velocidad Ventilador*



Sube el programa P5 Velocidad Ventilador a tú micro:bit (ver página 8). Conecta los componentes como se muestra a la derecha. Nota que realmente no importa el modo en el que se conecten los cables al motor. Inviértelos y verás como el motor girara en sentido contrario.

La pantalla de la micro:bit mostrará un número entre 0 y 9 que indica la velocidad de giro del ventilador. Usa el botón B (de la micro:bit) para incrementar la velocidad y el botón A para reducirla.

Cómo funciona

El proyecto usa el Relé MonkMakes para enviar pulsos de potencia de diferente longitud al motor.

El código es el más complicado hasta ahora. Usa unas pocas variables que son definidas en el bloque a*l Iniciar*.

- potencia_minima – Debería tener el valor de la mínima potencia para la cual el motor empieza a girar. Este valor varía entre diferentes motores. Prueba con el valor por defecto de 600, pero si el motor no gira cuando pulses el botón B para establecer la velocidad a 1, quizá tengas que incrementar este valor hasta 700 o más.
- potencia_maxima – Es la máxima potencia, puedes dejarlo con el valor de 1023
- potencia_paso – Es el valor del rango de potencia dividido por 9 y será la cantidad que cambia la potencia cada vez que incrementamos un paso la velocidad.

- velocidad – Es la velocidad entre 0 (apagado) a 9 (máximo)

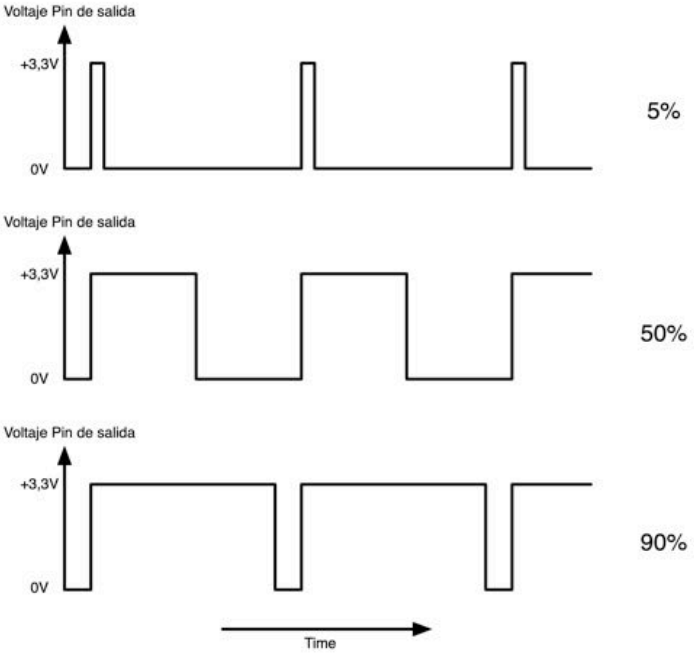
El bucle *para siempre* se usa para establecer de manera continua el nivel de potencia indicado por la variable velocidad. El caso en el que la variable velocidad es 0 se trata de una forma especial y ponemos el motor a potencia 0, si no lo hiciéramos así, el motor vibraría sin llegar a girar cuando la velocidad fuera 0.

El bloque *al presionarse el botón* es muy similar a los otros. El correspondiente al botón A resta 1 de la variable velocidad y se asegura que el valor de velocidad no sea inferior a 0, además muestra el valor de la nueva velocidad y usa el bloque *escritura analógica* para establecer la salida de potencia del pin 0. El bloque *al presionar el botón* B hace lo mismo pero añadiendo uno a la velocidad.

El método para controlar la potencia que usa *escritura analógica* se denomina Modulación de Ancho de Pulso o PWM (Pulse Width Modulation), y funciona variando la duración de una serie de pulsos que se envían al relé.

Si el pulso solamente está en estado alto por un corto periodo de tiempo (digamos el 5% del tiempo) entonces solo se está enviando una pequeña cantidad de

potencia al motor y este gira lentamente. Cuanto más tiempo esté el pulso en estado alto, mayor potencia se entrega el motor y más rápido será el giro.



Código MicroPython

Si quieres usar la versión MicroPython de los programas en lugar de los bloques en el Editor de Bloques, revisa la sección casi al final del libro llamada *Usando MicroPython* para instrucciones sobre cómo descargar y usar el código.

Este sería el código MicroPython para este proyecto.

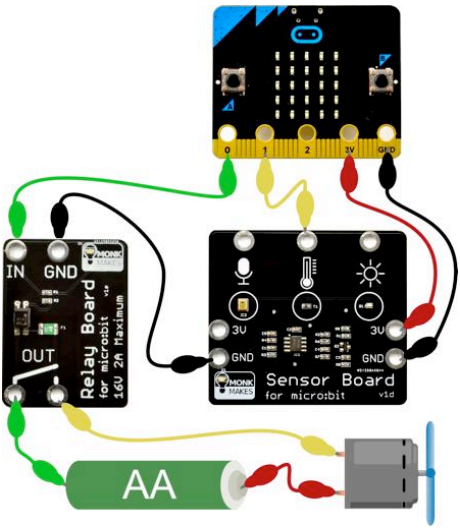
```
from microbit import *


potencia_minima = 600
potencia_maxima = 1023
potencia_paso = (potencia_maxima - potencia_minima) / 9
velocidad = 0
```

```
def establecer_potencia(velocidad):
	display.show(str(velocidad))
	if velocidad == 0:
		pin0.write_analog(0)
	else:
		pin0.write_analog(velocidad * potencia_paso +
	potencia_minima)

establecer_potencia(velocidad)

while True:
	if button_a.was_pressed():
		velocidad -= 1
		if velocidad < 0:
			velocidad = 0
		establecer_potencia(velocidad)
	elif button_b.was_pressed():
		velocidad += 1
		if velocidad > 9:
			velocidad = 9
		establecer_potencia(velocidad)
	sleep(100)
```

Proyecto 6. Ventilador Automático

Lo que necesitarás:

- micro:bit
- Placa de Sensores de MonkMakes para micro:bit
- Relé MonkMakes para micro:bit
- Pila AA y su caja portapilas
- Motor con la hélice colocada
- Programa: *P6 Ventilador automático* o su versión Fahrenheit *P6 Ventilador Automático F*



Sube el programa *P6 Ventilador automático* (para Centígrados) o *Ventilador Automático F (*para Fahrenheit*)* a tu micro:bit (mira la página 8). Conecta después los componentes como se ve a la derecha. Compruébalo cuando hayas terminado, este montaje es bastante complicado.

Este proyecto visualizará la temperatura en la pantalla y arrancará automáticamente el ventilador cuando la temperatura suba a partir de una temperatura establecida (*temp_umbral*). Cuando el programa arranca por primera vez el valor de *temp_umbral* se establece 2 grados por encima de la temperatura real . Puedes incrementar o decrementar el valor de *temp_umbral* en un grado pulsando el botón B o el botón A respectivamente

Para comprobar el funcionamiento del proyecto, establece el valor de *temp_umbral* 1 grado por encima de la temperatura actual y después pon tu dedo en la zona del sensor de temperatura de la placa de Sensores de MonkMakes para simular que la temperatura de la habitación ha aumentado. Cuando la temperatura suba por encima de la temperatura establecida el ventilador arrancará. Aparta tu dedo y después de unos segundos la temperatura del sensor habrá bajado por debajo de *temp_umbral* y el motor se apaga de nuevo

Cómo funciona

El código para la versión en grados centígrados de este proyecto es muy similar al termómetro del proyecto 4 excepto que existe una nueva variable llamada *temp_umbral* qué se almacena la temperatura deseada y en el bucle *para siempre* se comprueba si la temperatura actual está por encima de esta temperatura. Cuando esto ocurre se usa el bloque *escritura digital* para activar el relé y por tanto

el ventilador.

El bloque *al presionarse el botón* añade un grado a la variable *temp_umbral*

Código MicroPython

Si quieres usar la versión MicroPython de los programas en lugar de los bloques en el Editor de Bloques, revisa la sección casi al final del libro llamada *Usando MicroPython* para instrucciones sobre cómo descargar y usar el código.

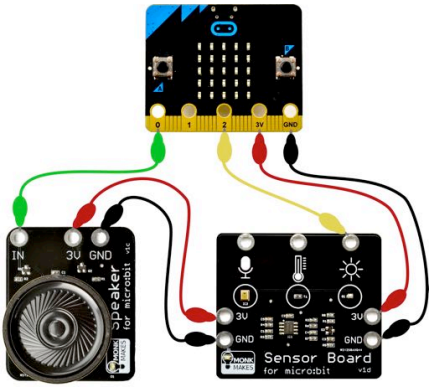
Este sería el código MicroPython para este proyecto.

```
from microbit import *


def lee_temp_c():
    lectura = pin1.read_analog()
    temp_c = round( lectura * 0.157 - 54)
    return temp_c


temp_umbral = lee_temp_c() + 2


while True:
    if button_a.was_pressed():
        temp_umbral -= 1
        display.scroll('Umbral' + str(temp_umbral))
    elif button_b.was_pressed():
        temp_umbral += 1
        display.scroll('Umbral' + str(temp_umbral))
    display.scroll(str(read_temp_c()))
    if lee_temp_c() > temp_umbral:
        pin0.write_digital(1)
    else:
        pin0.write_digital(0)
```

Proyecto 7. Música Mágica

Lo que necesitarás:

- micro:bit
- Placa de Sensores MonkMakes para micro:bit
- Altavoz MonkMakes para micro:bit
- Programa: *P7 Música Mágica*



Sube el programa *P7 Música Mágica* a tu micro:bit (see page 8). Después conecta como se ve a la derecha. Nota como la placa de sensores pasa la alimentación desde la micro:bit a el Altavoz.

Agita tus manos delante del sensor de luz y deberías oír sonidos interesantes.

Si tu lugar de trabajo está muy iluminado y no oyes ningún sonido, quizás necesites cambiar el valor de la variable *multiplicador* de 20 a 10 (verás el motivo en la siguiente sección).

Cómo funciona

Este es el código del proyecto.

La variable *multiplicador* tiene inicialmente un valor de 20. Se usa para incrementar el valor desde el nivel de luz hasta una frecuencia que sea audible. Por tanto, si cuando mueves tu mano delante del sensor de luz, todo lo que oyes son chasquidos, deberás incrementar su valor. Por otro lado, si cuando mueves tu mano cerca del sensor de luz, el tono del sonido se incrementa demasiado como para que sea audible, intenta reducir el valor de *multiplicador* a 10.

Cuando el programa arranca, lee el nivel de luz y lo guarda en la variable *base_iluminacion*. Por tanto, si mueves tu proyecto a una habitación más iluminada, debes pulsar el botón Reset de la micro:bit para que vuelva a tomar el valor de referencia para el brillo de la habitación.

Código MicroPython

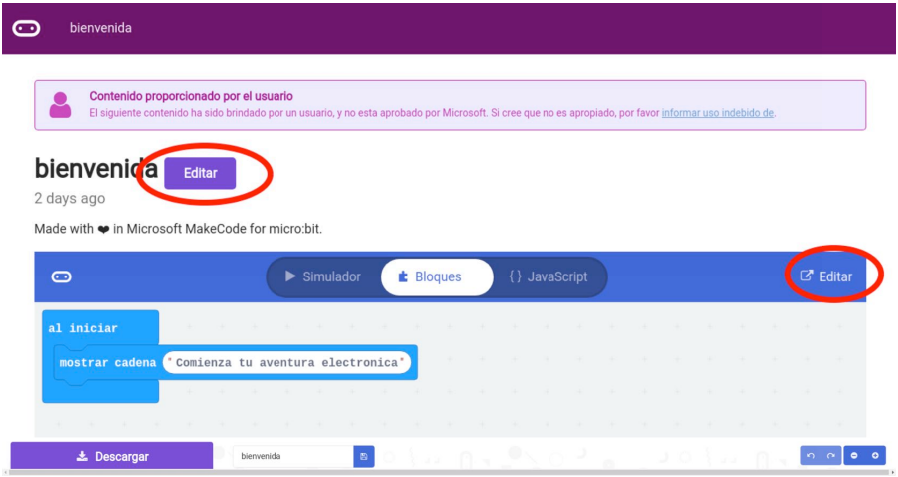
Si quieres usar la versión MicroPython de los programas en lugar de los bloques en el Editor de Bloques, revisa la sección casi al final del libro llamada *Usando MicroPython* para instrucciones sobre cómo descargar y usar el código.

Este sería el código MicroPython para este proyecto.

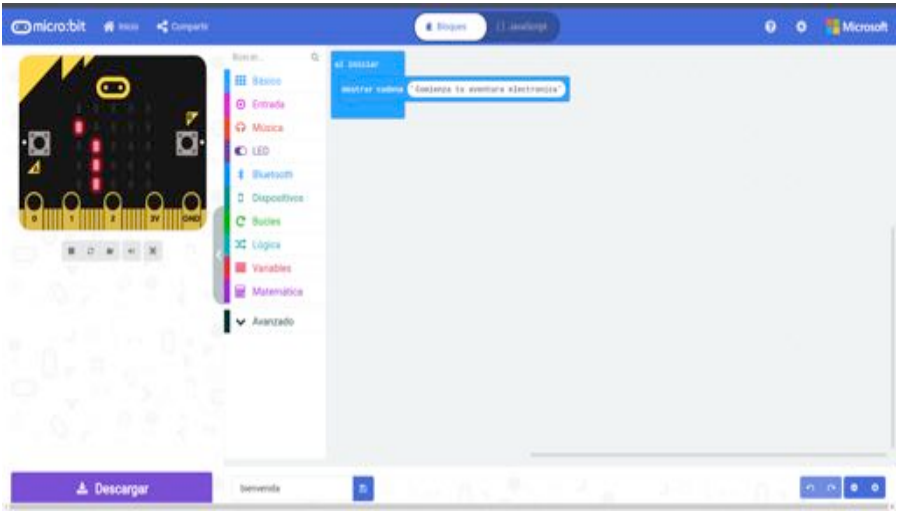
```
from microbit import *
import music

multiplicador = 20

base_iluminacion = pin2.read_analog()

while True:
    iluminacion = pin2.read_analog()
    tono = base_iluminacion - iluminacion
    if tono > 0:
        music.pitch(tono * multiplicador)
    sleep(100)
```

El Editor De Bloques de JavaScript

Si quieres modificar uno de los programas de bloques para kit, puedes hacerlo haciendo clic en cualquiera de los dos botones Editar .



Al pulsar cualquiera de los botones Editar se abrirá el proyecto en el Editor de Bloques.



Una característica interesante del Editor de Bloques es que la imagen de la placa micro:bit de la izquierda de la pantalla es en realidad una micro:bit virtual que puede ejecutar tus programas antes de que los descargues dentro de tu micro:bit real. Puedes presionar sus botones con el ratón. Mostrará cosas y si estás usando pines GPIO como salidas digitales las encenderá cuando tú las actives. También puedes hacer clic en los pines GPIO para simular entradas digitales y/o analógicas. La parte del centro de la pantalla tiene diferentes categorías de bloques: básicos, entradas, música, etc... dónde encontrarás los bloques que pondrás en la parte de la derecha, el área llamada lienzo. También puedes usar la caja de búsqueda, justo encima de la lista de las categorías de bloques, si no está seguro de dónde encontrar el bloque que quieres usar. El Editor de bloques realmente genera código escrito en un lenguaje de programación llamado JavaScript y si pulsas en el botón JavaScript puedes ver el código que se ha generado. Pulsando en bloques te devuelve a la vista de bloques en la cual es mucho más fácil trabajar.

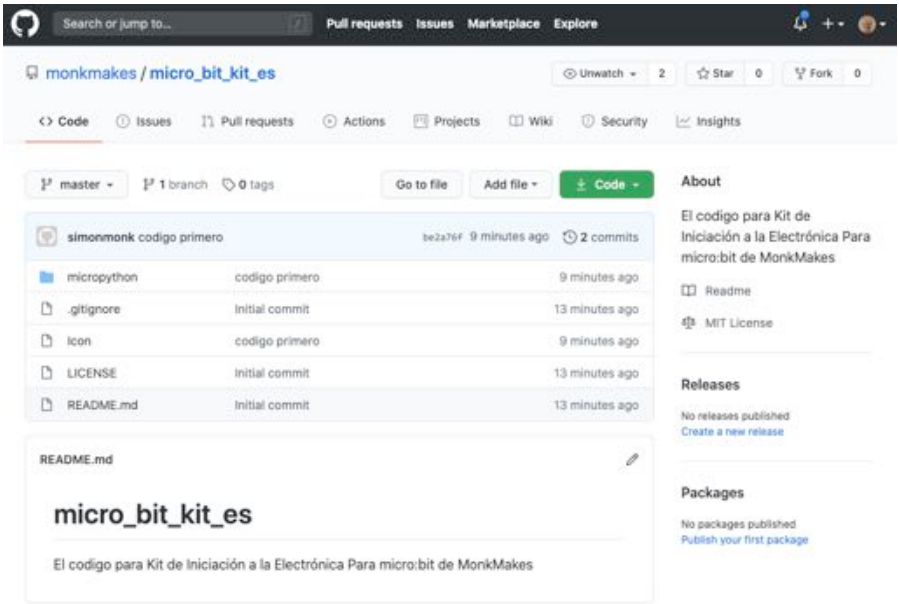
Eventos

El lenguaje JavaScript que usa internamente el Editor de Bloques utiliza lo que se llaman “eventos”. El bloque “al iniciar” donde aparecía nuestro mensaje de bienvenida es un ejemplo de evento. Este evento indica que la micro bit acaba de arrancar porque se ha encendido o porque se ha pulsado el botón de reset. Empieza un nuevo proyecto (pulsando desde *Inicio* en *Nuevo Proyecto*) y borra los bloques *al iniciar* y *para siempre*. Añade un bloque *al presionar un botón*, que es otro evento, desde la categoría *Entradas* y después coloca un bloque *mostrar texto* dentro y cambia el texto para que se muestre la cadena y se visualice algo parecido a lo siguiente

Ahora cuando hagas clic con tu ratón en el botón A en el micro:bit virtual aparecerá el mensaje. Puedes probarlo también en un micro:bit real.

Usando MicroPython

El código del kit está disponible tanto en bloques como en MicroPython. Si quieres ejecutar la versión Python de los programas en lugar de los bloques, puedes descargar todos los programas desde Github here: https://github.com/monkmakes/micro_bit_kit_es



Pulsa en el botón verde *Code* y selecciona la opción *Download ZIP*. Extrae el contenido del archivo ZIP y encontrarás una carpeta conteniendo los programas de todos los proyectos.

Puedes usar el editor Mu (https://codewith.mu/) para ver los programas descargados y subirlos a tu micro:bit.

Las versiones MicroPython de los programas son muy parecidas a sus equivalentes en Bloques y debería ser sencillo entenderlas.

Resolución de problemas

Problema: No puedo subir un programa a mi micro:bit

Solución: Primero comprueba que el LED de alimentación de la micro:bit está encendido. Si no es así prueba en otro puerto USB o con otro cable USB. Si la micro:bit recibe alimentación, pero no ves el disco micro:bit en tu ordenador, quizás estás usando un cable USB de 'sólo alimentación'. Prueba con otro cable.

Problema: El LED del Relé MonkMakes para micro:bit no se ilumina aunque creo que está conectado correctamente.

Solución: Es normal. El LED del Relé sólo se ilumina cuando el relé está activado.

Problema: Las medidas de temperatura en los proyectos no son muy precisas.

Solución: Limitaciones del sensor de temperatura y de los cálculos matemáticos de los Bloques dan lugar a que la temperatura medida sea sólo una aproximación. Revisa la sección sobre Calibración del proyecto 4 para hacer medidas más precisas.

Problema: El proyecto no funciona, pero estoy muy seguro de que he subido el programa correcto y el cableado parece correcto. ¿Qué puedo hacer?

Solución: Si has comprobado todo y no puedes encontrar el problema, es posible que uno de los cables o pinzas de cocodrilo esté rota – esto puede ocurrir si se dobla en exceso. Haz pruebas y piensa qué podría dar lugar al problema. Por ejemplo si el LED del logo del Sensor o del Altavoz no se enciende, inténtalo tras intercambiar los cables de alimentación (3V y GND) de la micro:bit.

Para otras preguntas, por favor comprueba la página web del producto (https://www.monkmakes.com/mb_kit_es/). También puedes enviar preguntas relacionadas con el soporte a support@monkmakes.com

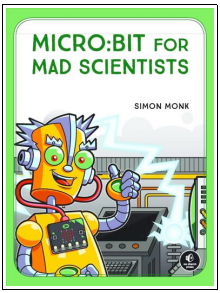
¿Qué Hacer para Seguir Avanzando?

Programando la micro:bit

Si quieres aprender más sobre cómo programar la micro:bit en Python y hablas ingles, podrías pensar en comprar el libro de Simon Monk 'Programming micro:bit: Getting Started with MicroPython', el cuál puedes encontrar en la principales librerías.

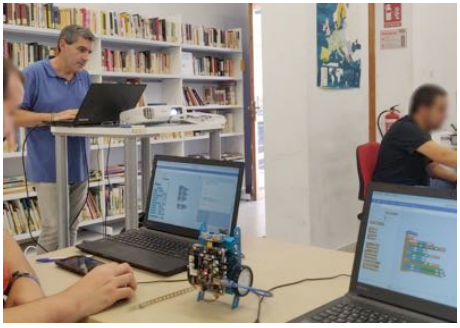
Para un libro de proyectos para micro:bit, revisa ‘micro:bit for Mad Scientists’. Ambos libros están disponibles en Amazon y en todas las buenas librerías.

Puedes encontrar más detalles sobre estos y otros libros de Simon Monk (el diseñador de este kit) en http://simonmonk.org o seguir su cuenta @simonmonk2 en Twitter



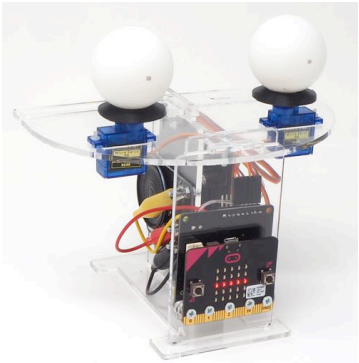
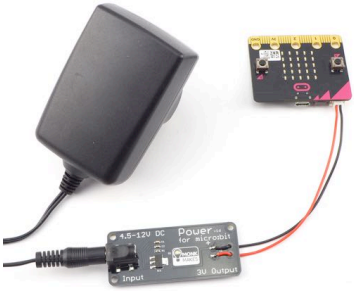
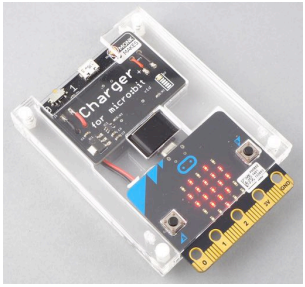
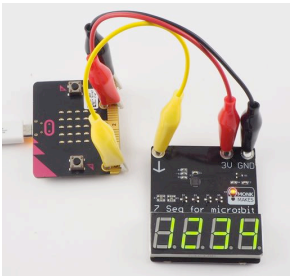
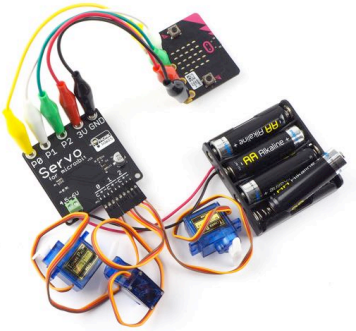
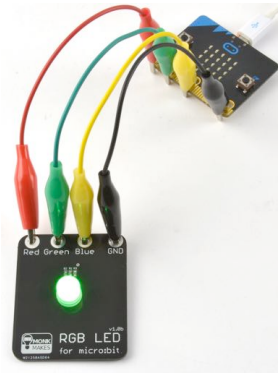
Jose Antonio Vacas

La traducción es de José Antonio Vacas. Jose, docente de temas de tecnología y robótica, (micro:bit, Arduino, Raspberry pi y diseño e impresión 3D...) Puede encontrar más información sobre sus cursos y otras actividades en su blog http://blog.elcacharreo.com/cursos/ y en su canal de youtube https://www.youtube.com/javacasm o seguirle en su twitter @javacasm



NOTAS

MonkMakes

Puedes consultar la página del producto para más información sobre este kit, https://monkmakes.com/mb_kit_es

Además de este kit, MonkMakes fabrica todo tipo de kits y accesorios que te ayudarán con tus proyectos con micro:bit y Raspberry Pi. Puedes consultar más detalles y ver dónde puedes comprarlos en https://monkmakes.com. También puedes serguir la cuenta de MonkMakes en Twitter @monkmakes .