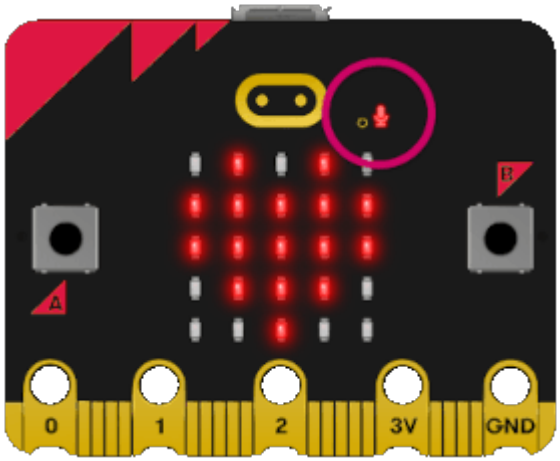
Micrófono

> Página extraída de Federico Coca Guia de Trabajo de Microbit CC-BY-SA

> ATENCIÓN SÓLO VÁLIDO PARA PLACAS V2

Este objeto permite acceder al micrófono integrado disponible en micro:bit V2. Se puede utilizar para responder al sonido. La entrada del micrófono se encuentra en la parte frontal de la placa junto a un LED de actividad del micrófono, que se ilumina cuando el micrófono está en uso.



Autor Federico Coca Fuente : Guía de Trabajo de Microbit Licencia CC-BY-SA

El micrófono puede responder a un conjunto predefinido de **eventos sonoros** que se basan en la amplitud y la longitud de onda del sonido. Están representados por instancias de la clase `SoundEvent`, accesibles a través de variables en `microbit.SoundEvent`:

- `microbit.SoundEvent.QUIET`: Representa la transición de eventos de sonido, de fuerte (`loud`) a silencioso (`quiet`) como hablar tranquilo o música de fondo a bajo volumen.
- `microbit.SoundEvent.LOUD`: Representa la transición de eventos de sonido, de silencioso (`quiet`) a fuerte (`loud`) como aplausos o hablar a gritos.

Las funciones disponibles son:

- `microbit.microphone.current_event()`: Retorna el nombre del último evento sonoro grabado, `SoundEvent('loud')` o `SoundEvent('quiet')`.
- `microbit.microphone.was_event(event)`: donde `event` es un evento sonoro como `SoundEvent.LOUD` o `SoundEvent.QUIET`. Retorna `true` si el sonido se ha escuchado al menos una vez desde la última llamada, en caso contrario `false`. `was_event()` también borra el historial de eventos de sonido antes de retornar.
- `microbit.microphone.is_event(event)`: donde `event` es un evento sonoro como `SoundEvent.LOUD` o `SoundEvent.QUIET`. Retorna `true` si el evento sonoro es el más reciente desde la última llamada, en caso contrario `false`. No borra el historial de eventos de sonido.
- `microbit.microphone.get_events()`: Retorna una tupla del historial de eventos. El más reciente aparece en último lugar. `get_events()` también borra el historial de eventos de sonido antes de retornar.
- `microbit.microphone.set_threshold(event, value)`: donde `event` es un evento sonoro como `SoundEvent.LOUD` o `SoundEvent.QUIET`. `value` es el umbral en el rango 0-255. Por ejemplo `set_threshold(SoundEvent.LOUD, 250)` sólo se activará si el sonido es muy alto (>= 250).
- `microbit.microphone.sound_level()`: Retorna una representación del nivel de presión sonora en el intervalo de 0 a 255.

Sonómetro

No estaría nada mal poner esto en clase, comedores...

```
from microbit import *

# definicion funcion mapea para cambiar un rango de valores a otro
def mapea(valor, deMin, deMax, aMin, aMax):
    deRango = deMax - deMin
    aRango = aMax - aMin
    valorEsc_de = float(valor - deMin)/float(deRango)
    valorEsc_a = aMin + (valorEsc_de * aRango)
    return valorEsc_a

# Creamos los imagenes para el grafico de barras
grafico5 = Image("99999:"
                 "99999:"
                 "99999:"
                 "99999:"
                 "99999")
```

```
grafico4 = Image("00000:"
                 "99999:"
                 "99999:"
                 "99999:"
                 "99999")

grafico3 = Image("00000:"
                 "00000:"
                 "99999:"
                 "99999:"
                 "99999")

grafico2 = Image("00000:"
                 "00000:"
                 "00000:"
                 "99999:"
                 "99999")

grafico1 = Image("00000:"
                 "00000:"
                 "00000:"
                 "00000:"
                 "99999")

grafico0 = Image("00000:"
                 "00000:"
                 "00000:"
                 "00000:"
                 "00000")

graficos = [grafico0, grafico1, grafico2, grafico3, grafico4, grafico5]

# ignora el primer nivel de sonido leido
nivelSonido = microphone.sound_level()
sleep(200)
```

Copyright 2026

```
# establece un umbral para el nivel de sonido
umbral = microphone.set_threshold(SoundEvent.LOUD, 125)
while True:
    # si el umbral es superado se muestra una carita triste
    if microphone.sound_level() >= 125:
        display.show(Image.SAD)
        sleep(1000)
    else:
        # mapear nivel de sonido de 0-255 a 0-5 para escoger gráfico
        nivelSonido = int(mapea(microphone.sound_level(), 0, 255, 0, 5))
        display.show(graficos[nivelSonido])
```

https://www.youtube.com/embed/IGcbk1rR8FQ

Uso de las API

Un ejemplo que utiliza algunas de las funciones de la API del micrófono es:

```
'''Prueba básica del micrófono.
Boton A: actualizar pantalla cuando se escucha un sonido alto o bajo.
Botón B: actualizar la pantalla cuando se escucho un sonido alto o bajo.
Al agitarla: se muestran los últimos sonidos escuchados, para intentar esta prueba
se hace un sonido fuerte y uno silencioso antes de agitar.'''

from microbit import *

display.clear()
sound = microphone.current_event()

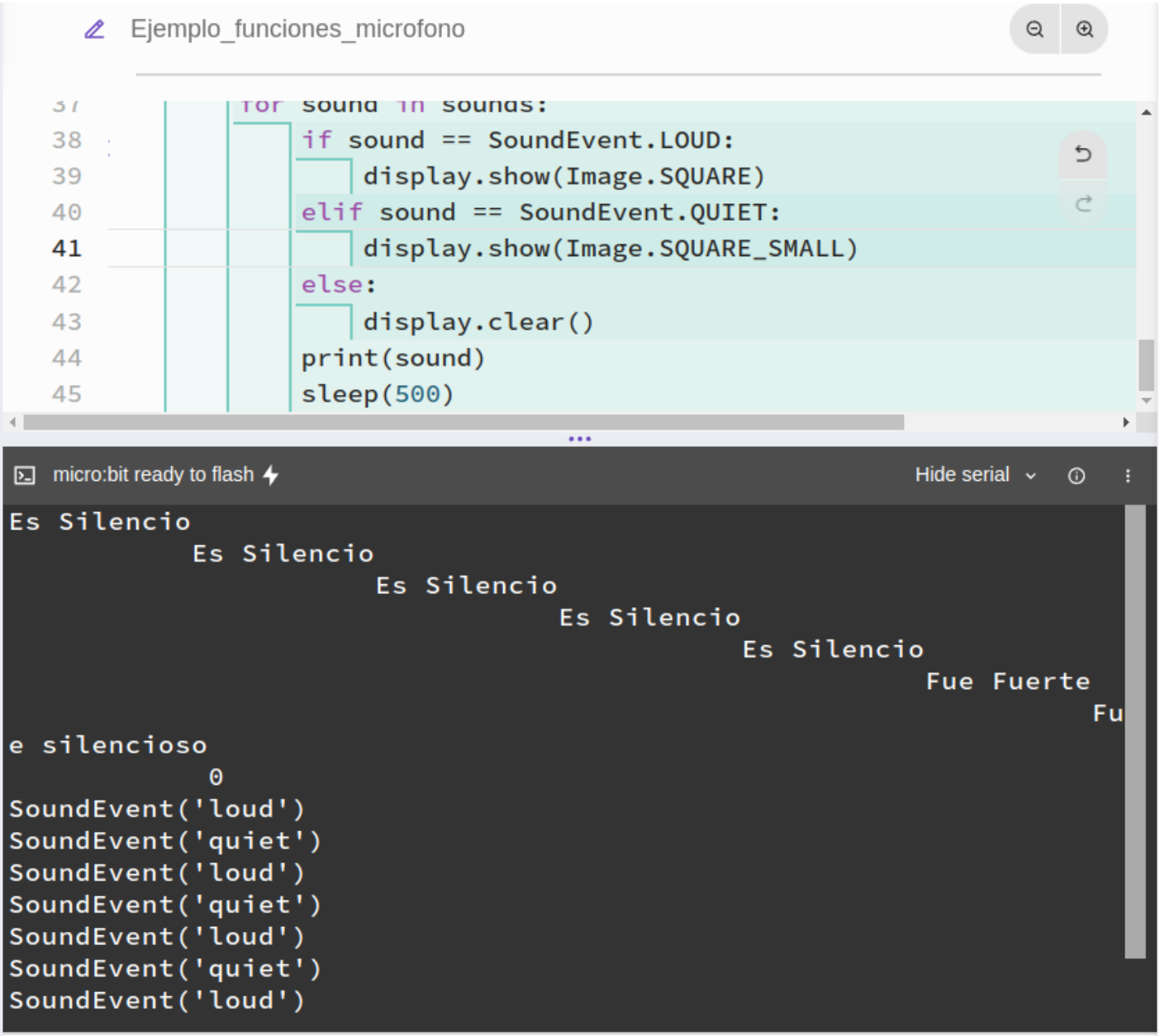
while True:
    if button_a.is_pressed():
        if microphone.current_event() == SoundEvent.LOUD:
            display.show(Image.SQUARE)
            uart.write('Es Fuerte\n')
        elif microphone.current_event() == SoundEvent.QUIET:
```

Copyright 2026

```
            display.show(Image.SQUARE_SMALL)
            uart.write('Es Silencio\n')
        sleep(500)
    display.clear()
    if button_b.is_pressed():
        if microphone.was_event(SoundEvent.LOUD):
            display.show(Image.SQUARE)
            uart.write('Fue Fuerte\n')
        elif microphone.was_event(SoundEvent.QUIET):
            display.show(Image.SQUARE_SMALL)
            uart.write('Fue silencioso\n')
        else:
            display.clear()
        sleep(500)
    display.clear()
    if accelerometer.was_gesture('shake'):
        sounds = microphone.get_events()
        soundLevel = microphone.sound_level()
        print(soundLevel)
        for sound in sounds:
            if sound == SoundEvent.LOUD:
                display.show(Image.SQUARE)
            elif sound == SoundEvent.QUIET:
                display.show(Image.SQUARE_SMALL)
            else:
                display.clear()
            print(sound)
            sleep(500)
```

En la consola serie vemos algunos resultados:

Copyright 2026



Autor Federico Coca Fuente : Guía de Trabajo de Microbit Licencia CC-BY-SA

Se ve mejor con un vídeo, pero con el simulador que nos muestra la cantidad de sonido:

- El botón A muestra el sonido presente
- El botón B muestra los sonidos pasados
- Sacudir la microbit nos muestra en mensaje y numero el sonido presente

https://www.youtube.com/embed/EWnYdzFa1nM

Página extraída de Federico Coca Guia de Trabajo de Microbit CC-BY-SA

Revision #9
Created 2024-09-27 12:26:36 CEST by Javier Quintana
Updated 2024-10-01 19:42:14 CEST by Javier Quintana