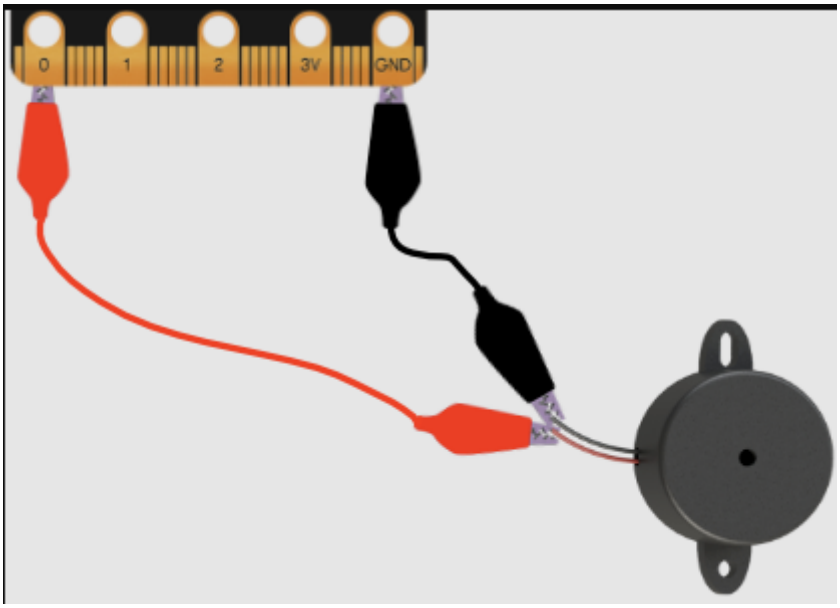
Musica

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

MicroPython de BBC micro:bit viene acompañado de un potente módulo de música y sonido. Es muy fácil generar pitidos y zumbidos desde el dispositivo conectando un altavoz o unos auriculares con cable, o utilizando el altavoz integrado si estamos con una versión V2.

La forma de conectar unos auriculares está descrita en el apartado de MakeCode. También se puede conectar un zumbador piezoeléctrico pasivo o un altavoz con pinzas de cocodrilo. Estos elementos pueden estar polarizados por lo que tendremos que comprobar si existe un terminal "+", y si es así conectar al pin0.



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

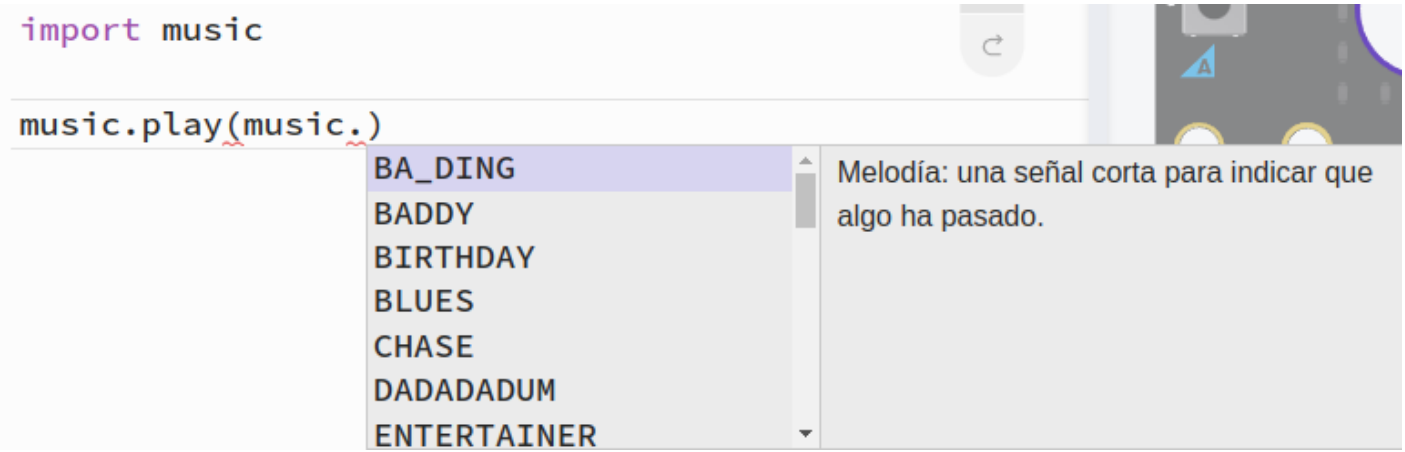
> **Nota:** Debemos asegurarnos de que el zumbador es pasivo y no activo, que tan solo puede reproducir un tono. En el apartado 'Componentes discretos' de Conceptos técnicos podemos encontrar como distinguirlos.

Para trabajar con música hacemos:

```
import music
music.play(music.NYAN)
```

Tenemos que importar módulo `music` que contiene los métodos para crear y controlar el sonido.

La función de autocompletado de MicroPython nos muestra las melodías incorporadas.



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

En la referencia de la API podemos encontrar mas inforamción en inglés sobre Music

Cada nota tiene un nombre como Do# (C#) o Fa (F), una octava, que indica lo alta o baja que debe tocarse y una duración. Las octavas se indican con un número, siendo 0 la octava más baja, 4 la del Do central y 8 es la más alta. Las duraciones también se expresan con números. Estos valores están relacionados entre sí: por ejemplo, una duración de 4 es el doble que una duración de 2 (y así sucesivamente). Si utilizamos como nombre de nota `R`, MicroPython reproducirá un silencio de la duración especificada.

Cada nota se expresa como una cadena de caracteres como ésta:

```
Nombre_nota[octave][:duration] #La1:4 (A1:4) es un La en la octava 1 con una duración de 4
```

Crear listas de notas para hacer una melodia es similar a crear una animación con una lista de imágenes. En el ejemplo vemos como sería la apertura de "Frere Jaques":

```
import music

frere_jaques_o = ["C4:4", "D4:4", "E4:4", "C4:4", "C4:4", "D4:4", "E4:4", "C4:4",
```

```
        "E4:4", "F4:4", "G4:8", "E4:4", "F4:4", "G4:8"]
music.play(frere_jaques_o)
```

El ejemplo se puede re-escribir como vemos a continuación ya que los valores de octava y duración se rpiten hasta que se indique un cambio.

```
import music

frere_jaques_o = ["C4:4", "D", "E", "C", "C", "D", "E", "C", "E", "F", "G:8",
        "E:4", "F", "G:8"]
music.play(frere_jaques_o)
```

MicroPython nos permite crear tonos que no son notas musicales. Por ejemplo, este código crea un efecto de sirena de policía:

```
import music

while True:
    for frecuencia in range(880, 1760, 16):
        music.pitch(frecuencia, 6)
    for frecuencia in range(1760, 880, -16):
        music.pitch(frecuencia, 6)
```

El método `music.pitch` utiliza una frecuencia que puede ser la de una nota musical. En los rangos de `frecuencia` se especifican los tonos de los sonidos de una sirena como "valor inicial, valor final y paso". Cuando el paso en positivo sube el tono y cuando es negativo lo baja.

El ejemplo también nos muestra como anidar distintos tipos de bucle.

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

Revision #1
Created 2024-09-26 13:34:27 CEST by Javier Quintana
Updated 2024-09-26 13:36:56 CEST by Javier Quintana