Variables, lógica y funciones

- Variables y arreglos
- Dado de seis caras
- Metrónomo digital
- Uso de funciones

Variables y arreglos

> Una variable es un dato almacenado en la memoria del ordenador, al que nos referimos por un nombre y cuyo valor puede ser cambiado durante la ejecución de un programa.

Las variables pueden almacenar tres tipos básicos de datos:

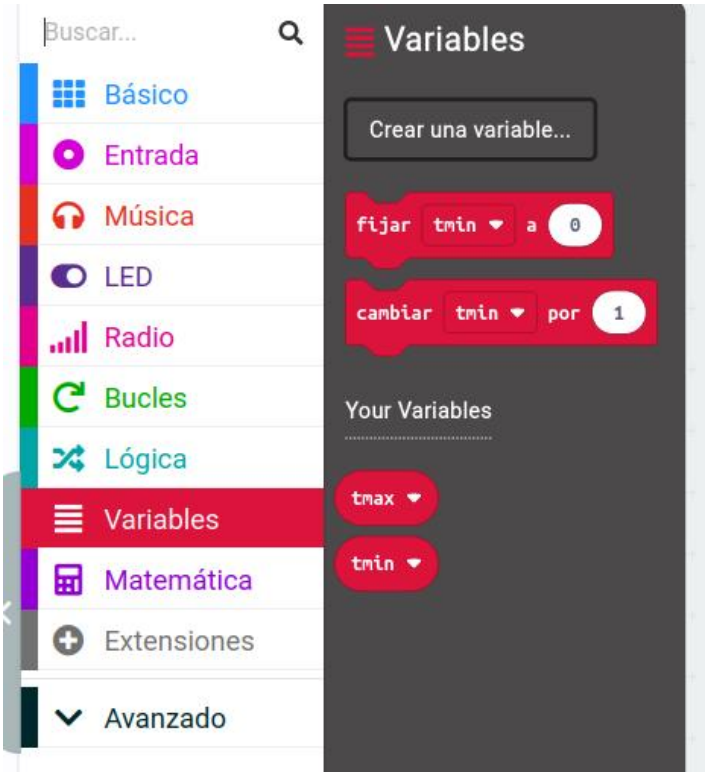
- **Booleanos**. Un dato de este tipo sólo puede tomar los valores verdadero o falso, a veces nombrados como true y false, high y low, alto y bajo ó 0 y 1.
- **Numéricos**. Los números almacenados pueden ser enteros, reales, complejos, etc.
- **Alfanuméricos**, como caracteres o cadenas de texto

Para ilustrar el uso y la utilidad de las variables, vamos a modificar nuestro **termómetro digital** para que, además de la temperatura actual, muestre también las **temperaturas máxima y mínima** registradas a lo largo del tiempo.

Comenzaremos por la parte más básica: al pulsar el botón A, micro:bit mostrará en pantalla la temperatura actual. Para ello usamos el bucle de eventos **al presionarse el botón A**, el bloque **mostrar número** y la función **-** para restar **3** grados a la lectura del sensor de **temperatura (ºC)** y obtener así una lectura más precisa, tal y como se indicó en el capítulo de **Bucles y eventos de tiempo: un termómetro**.

Las temperaturas **máxima** y **mínima** van a ser almacenadas en **dos variables** que debemos crear previamente. Puesto que cada variable debe tener asignado un nombre, vamos a llamarlas **tmax** y **tmin**.

Para crear una variable hay que deplegar el menú **Variables**, pulsar sobre **Crear un variable...** y darle el nombre deseado. Tras crear las variables **tmax** y **tmin**, el aspecto del menú debería ser el siguiente:



Nada más encender o reinicializar la placa conviene **guardar el valor de la temperatura actual en las variables creadas**. Para ello usaremos la estructura **al iniciar**, el valor proporcionado por la entrada **temperatura (ºC)** y las funciones de asignación de valores **fijar tmax a** y **fijar tmin a** .

> El bloque **fijar...a** asigna a la variable el valor numérico indicado, mientras que el bloque **cambiar... por** suma a la variable el valor indicado.

Las temperaturas máximas y mínimas guardadas en **tmax** y en **tmin** van a ser actualizadas cada minuto. Es necesario por lo tanto usar un evento de tiempo del tipo **cada 60000 ms** de la siguiente forma:

Hemos vuelto a usar dos bloques **fijar a** para cambiar los valores de las temperaturas máxima y mínima. Por otro lado, las funciones **min de...y...** y **max de...y...** se encuentran en el menú **Matemática**.

El primer bloque de asignación da a la variable **tmin** el valor mínimo entre la temperatura actualmente medida con la entrada **temperatura (º)** y la temperatura mínima anteriormente guardada en **tmin**.

El segundo bloque de asignación da a la variable **tmax** el valor máximo entre la temperatura actualmente medida con la entrada **temperatura (º)** y la temperatura máxima anteriormente guardada en **tmax**.

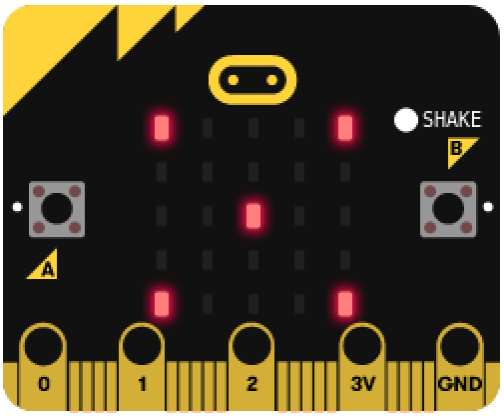
Ya sólo queda mostrar las temperaturas registradas cada vez que se pulse del botón B. No nos olvidarnos de restar 3 grados a las temperaturas guardadas con el fin de proporcionar unas lecturas más precisas.

Un **arreglo** es una variable que contiene varios valores numéricos o textuales organizados en filas. El menú **Arreglos** contiene bloques para crear arreglos, asignarles valores y acceder a los valores guardados. En la imagen se muestra un arpegio guardado como una lista de cuatro notas musicales.

Por último, Para manejar **variables de texto**, existen muchos bloques específicos disponible en el menú **Texto**.

Dado de seis caras

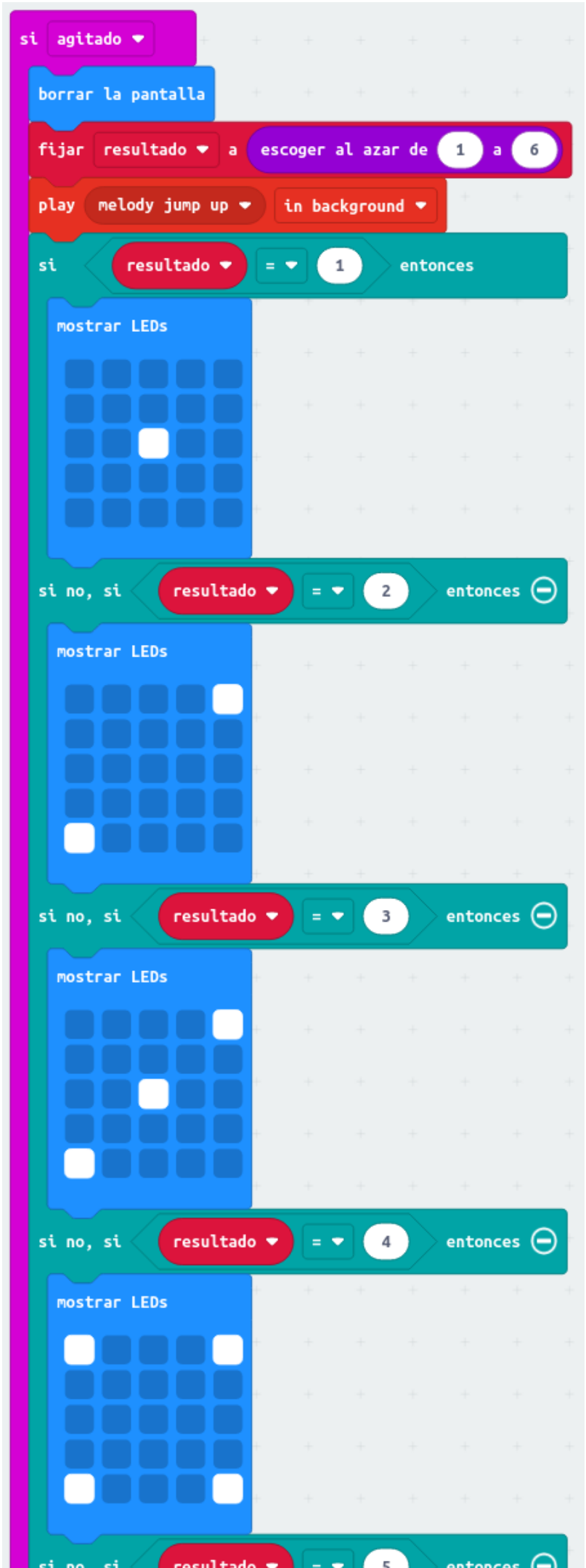
Mediante el generador de números aleatorios podemos convertir micro:bit en un **dado**, de forma que cada vez que la placa sea agitada muestre en pantalla el resultado de una tirada.



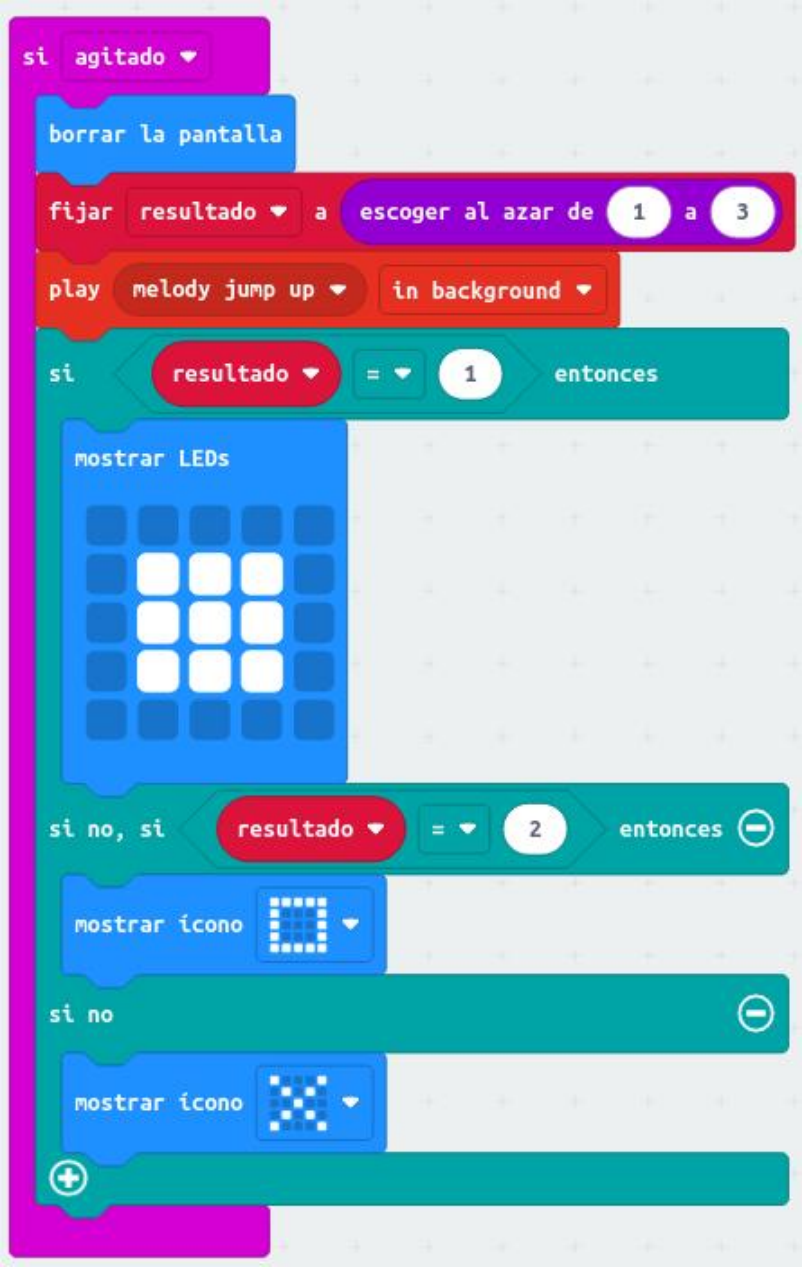
Para empezar, es preciso crear una variable que almacene el número aleatorio generado por micro:bit cuando sea agitado. Llamaremos a esta variable **resultado**, y la crearemos desde el menú **Variables**, para lo cual hay que pulsar sobre **Crear una variable...**

Un evento de tipo **si agitado** borrará la pantalla y hará sonar una pequeña melodía antes de mostrar el resultado de la tirada. El bloque **fijar resultado a escoger al azar de 1 a 6** asignará a la variable **resultado** un valor aleatorio de 1 a 6. El bloque **escoger al azar de...a...** se encuentra en el menú **Matemática**.

Seguidamente, una estructura condicional **si...entonces** decidirá qué icono mostrar en la pantalla en función del resultado.


si agitado
borrar la pantalla
fijar resultado a escoger al azar de 1 a 6
play melody jump up in background
si resultado = 1 entonces
mostrar LEDs
si no, si resultado = 2 entonces
mostrar LEDs
si no, si resultado = 3 entonces
mostrar LEDs
si no, si resultado = 4 entonces
mostrar LEDs

El código mostrado es muy fácil de adaptar a otros eventos aleatorios, como el lanzamiento de una moneda al aire o el juego de piedra, papel y tijera, cuyo código se presenta a continuación:



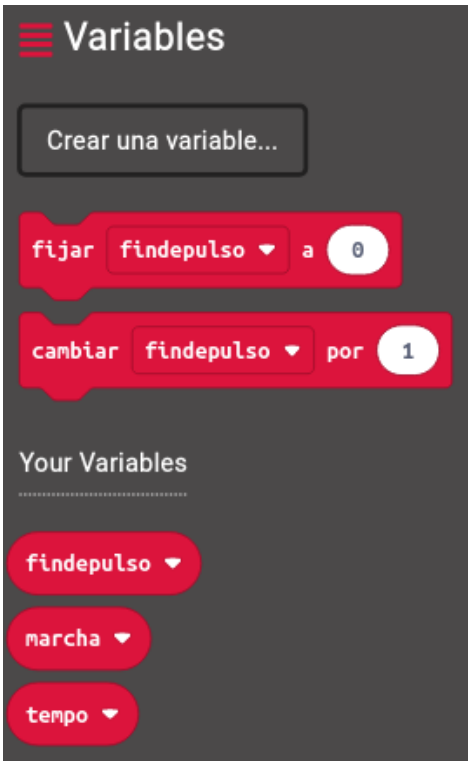
Metrónomo digital

Usaremos variables para convertir micro:bit en un metrónomo digital. Mediante los botones A y B cambiaremos el tempo del metrónomo, mientras que agitando la tarjeta encenderemos y apagaremos el sonido del metrónomo.

En primer necesitamos una variable que almacene el tempo, y a la que vamos a llamar, obviamente, **tempo**. Otra variable se llamará **marcha**, y almacenará los valores **verdadero**, si el metrónomo está en marcha o **falso**, si está parado. **Marcha** será por lo tanto una **variable de tipo booleano**.

La variable **findepulso** almacenará el momento exacto medido en milisegundos en el que debe acabar cada pulso.

Para crear las variables hay que acceder al menú **Variables** y pulsar sobre **Crear una variable...** El aspecto que presentará el menú desplegable una vez creadas las variables es el siguiente:



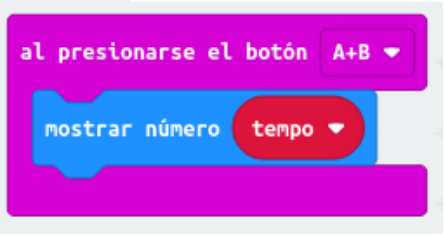
Seguidamente daremos unos valores iniciales a las variables: 120 para el **tempo** y **falso** para **marcha**. Este último valor indica que el metrónomo estará parado cuando sea encendido o reiniciado.

Recordemos que el bloque con la constante **falso** se obtiene del menú **Lógica**.

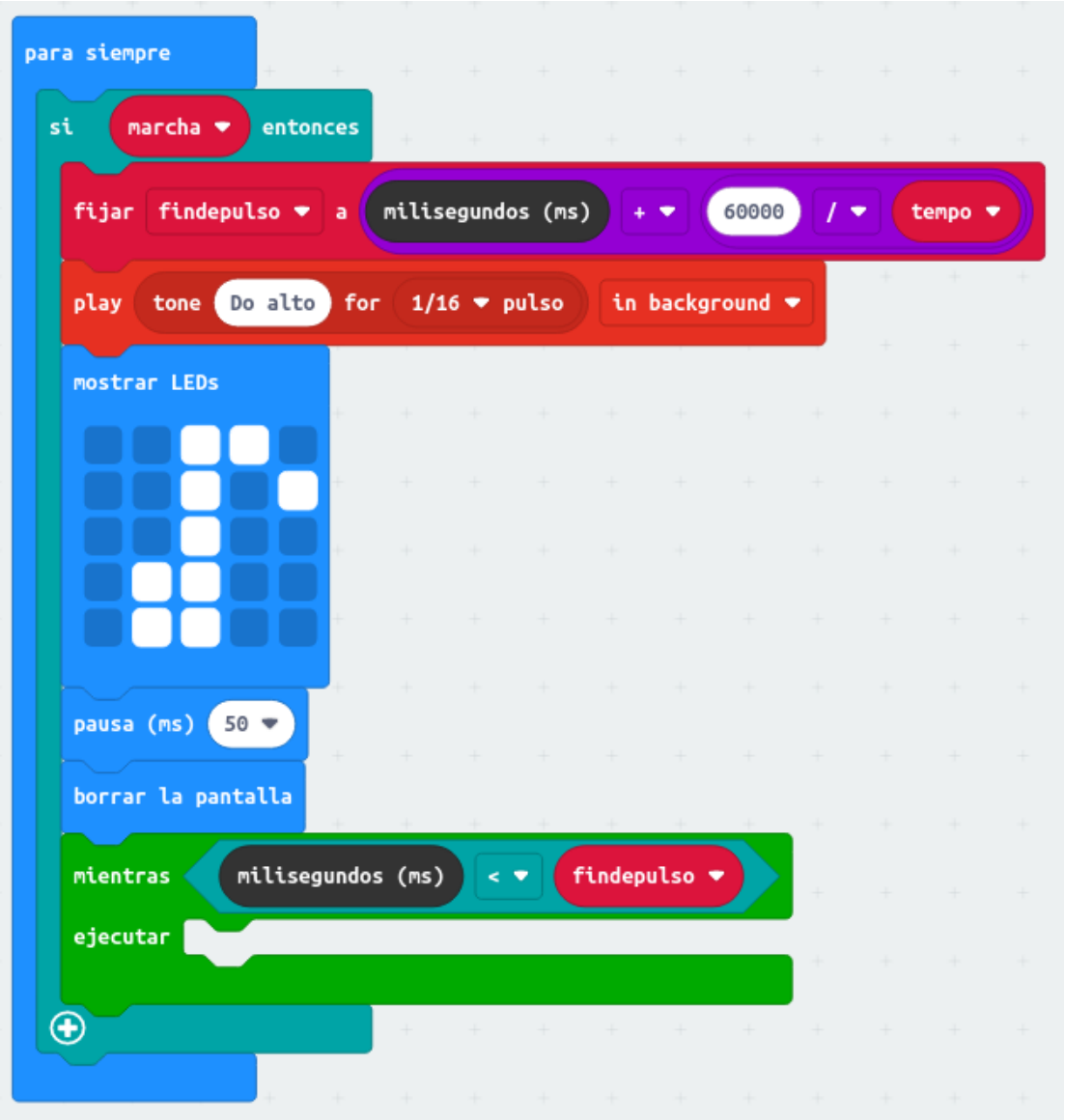
El metrónomo se pondrá en marcha o se parará cada vez que se agite. Usaremos un evento **si agitado** del menú **Entrada** y cambiaremos el valor de **marcha**. Gracias al operador **no**, del menú **Lógica**, si el valor de **marcha** es **falso**, cambiará a **verdadero**, y si es **verdadero**, cambiará a **falso**.

Cuando se pulse el botón A el tempo disminuirá un pulso y cuando se pulse el botón B el tempo aumentará un pulso.

Al pulsar A+B micro:bit mostrará en pantalla el tempo.



El bloque **para siempre** será el encargado de hacer sonar un pitido con cada pulso y de mostrar brevemente el icono de una corchea en la pantalla **LED**.



En primer lugar, el bucle **si marcha entonces** se asegura de que el metrónomo sólo funcione si la variable **marcha** tiene el valor **verdadero**. En caso contrario el bucle no hará nada.

Si el metrónomo está en marcha, la variable **findepulso** tomará el valor en milisegundos en el que debe acabar la ejecución del pulso actual. Este valor es la suma del tiempo actual, **milisegundos (ms)**, y de la duración de un pulso en milisegundos, es decir, 60000 ms/min dividido entre el tempo en pulsos/min. La variable **milisegundos (ms)** se encuentra en el menú **Avanzado/Control**.

Seguidamente se hace sonar una nota breve, se muestra una corchea en pantalla, se espera un instante y se borra la pantalla.

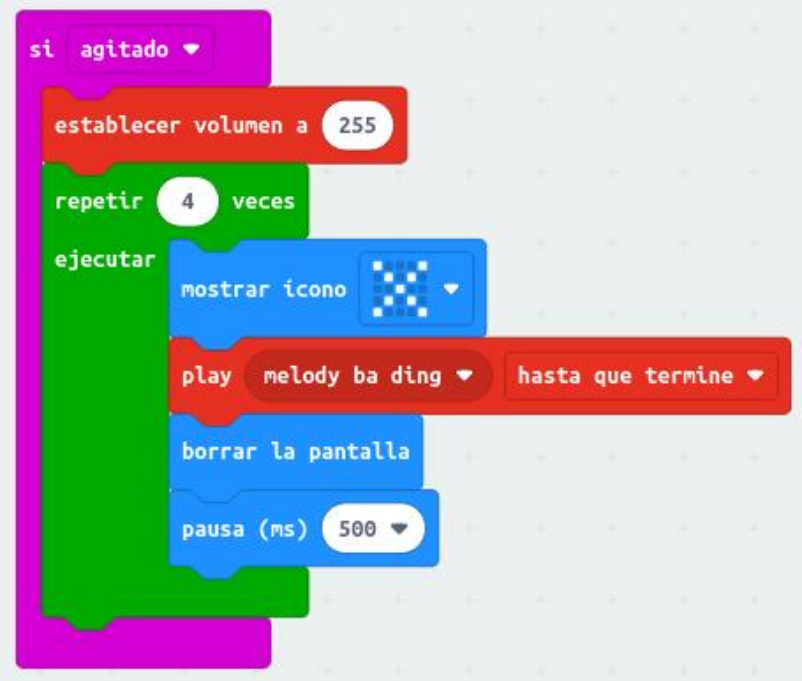
Como todavía quedará tiempo para que finalice el pulso actual, se introduce un bucle de espera cuyo funcionamiento es el siguiente: **mientras** el tiempo actual en ms sea menor que el tiempo en el que debe finalizar el pulso, no se hará nada. Por el contrario, cuando el tiempo actual en ms sea igual o mayor al tiempo de **findepulso**, el pulso habrá acabado, el bucle **mientras** finalizará y se iniciará una nueva ejecución del bucle **para siempre** (un nuevo pulso).

Uso de funciones

A menudo es preciso **repetir muchas veces a** lo largo de un programa **un mismo fragmento de código**. Para no tener que escribir el mismo código una y otra vez en aquellas partes del programa donde deba ser ejecutado, MakeCode permite el uso de **funciones**.

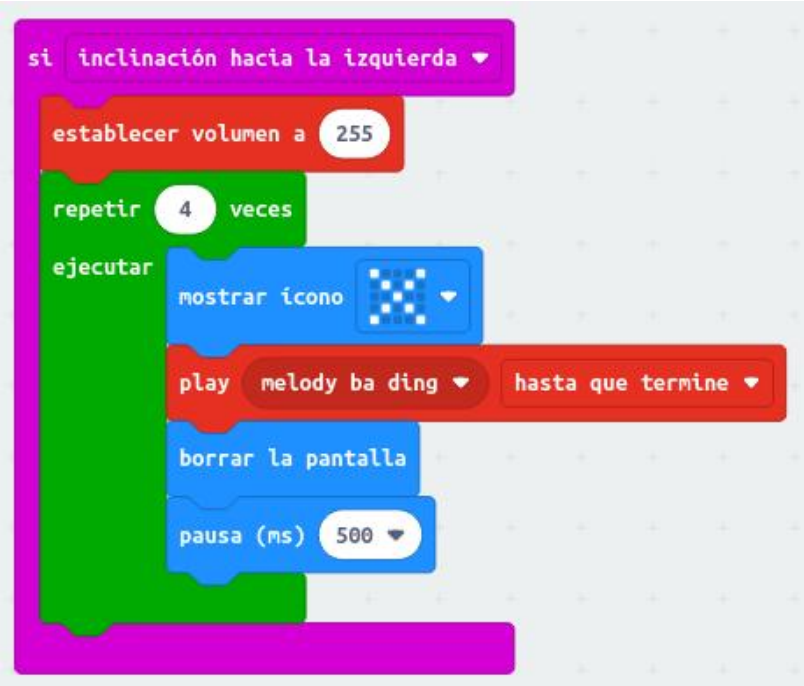
Una **función** es por lo tanto un trozo de código al que asignamos **un nombre que podemos invocar** en cualquier momento y desde cualquier parte del programa para que se ejecute. En otros lenguajes de programación, las funciones como las de MakeCode son denominidas **rutinas**, **subrutinas**, **módulos**, **métodos** o **procedimientos**.

Consideremos del código siguiente, que convierte a micro:bit en una **alarma**: cada vez que la tarjeta es agitada presenta un icono intermitente y produce un sonido.



Después de ajustar el volumen al máximo, un bucle **repetir 4 veces**, muestra un icono, después reproduce una melodía pregrabada por medio del bloque **play...hasta que termine**, borra la pantalla y espera medio segundo.

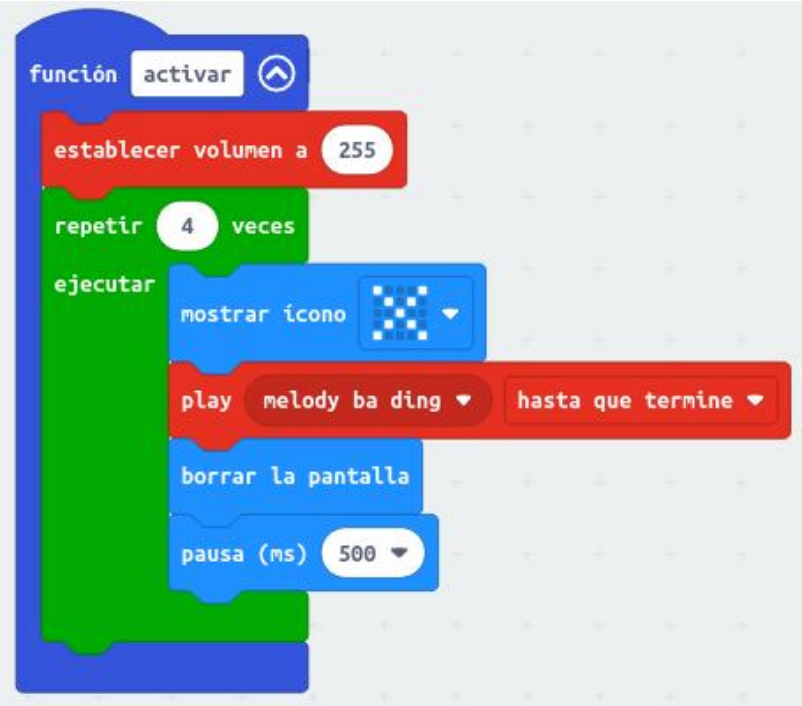
Podemos hacer más sensible la alarma añadiendo más eventos para dispararla, por ejemplo, el giro de la placa a la izquierda.



Para cualquier otro evento que deba activar la alarma habrá que repetir **volver a repetir el código**, llenando el área de programación de bloques. Solucionemos este probllema creando una función a la que vamos a llamar **activar**.

Tan sólo hay que pulsar en el menú **Avanzado** **Funciones**, pulsar sobre **crear una variable...** y darle el nombre elegido. Inmediatamente aparecerá el nuevo bloque de código correspondiente a la función activar.

Ahora debemos completar el código de la función:



Finalmente, cada evento que deba disparar la alarma incorporará una llamada a la función, que aparecerá como un bloque denominado como **llamada activar** dentro del menú **Funciones.**