Juego del gato y el ratón

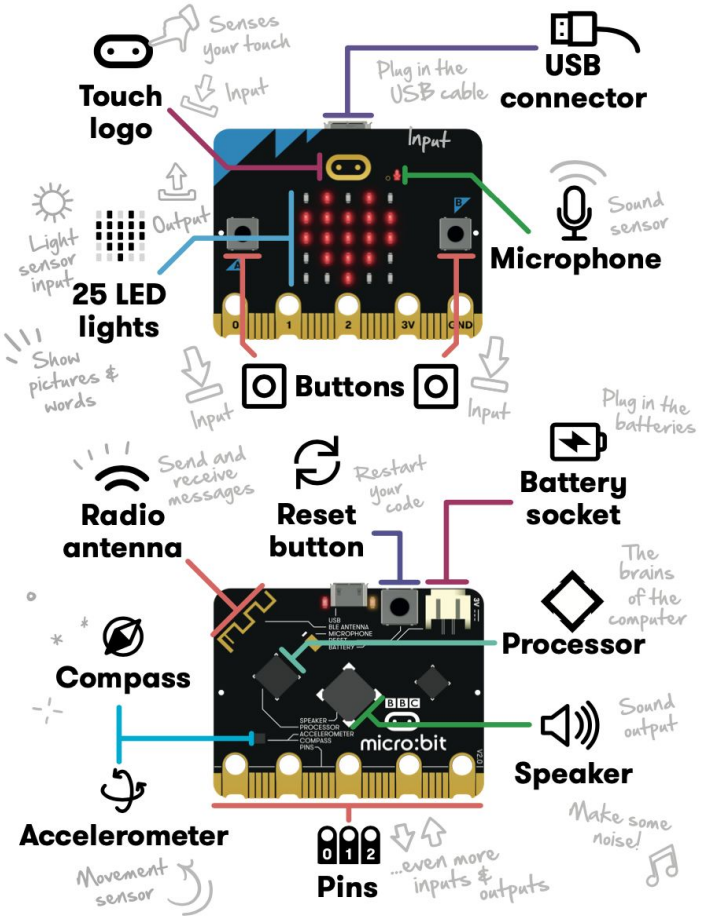
ZER Moianès Llevant 2023

PRESENTACIÓN

La placa **micro:bit** permite crear letras, números y dibujos de una forma sencilla.

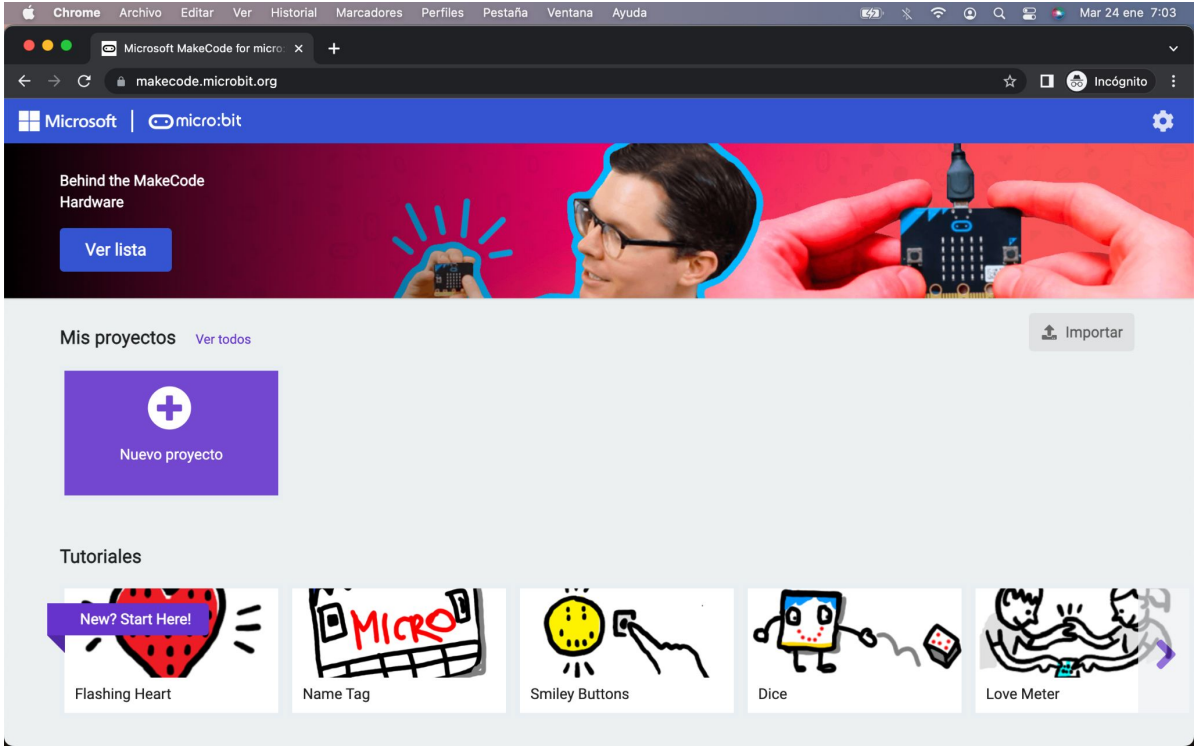
Vamos a intentar hacer un juego con **micro:bit**, el gato que persigue al ratón.

Veamos como hacerlo en el simulador de Makecode y cómo subirlo al **micro:bit**



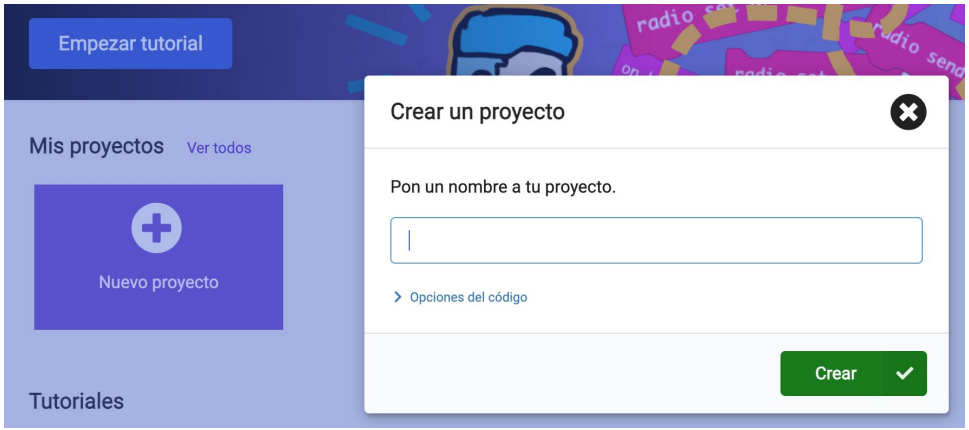
Empezando con MakeCode

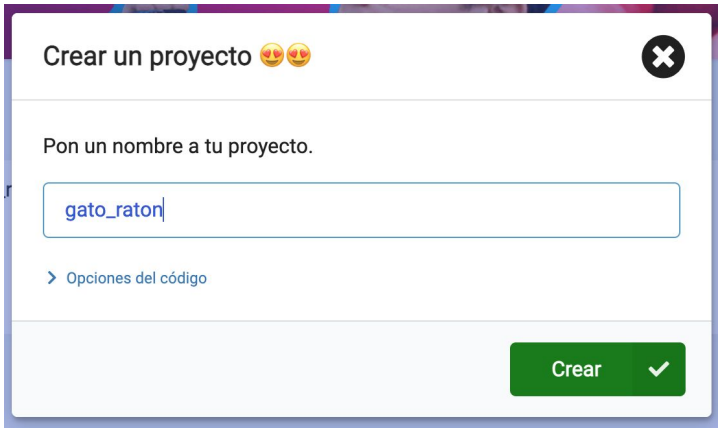
Escribimos en el navegador **makecode.microbit.org**



Nuevo proyecto MakeCode

Si pulsamos en “**Nuevo proyecto**” podremos dar un nombre a nuestro proyecto





Panel de trabajo y simulador de MakeCode

Opciones

Simulador

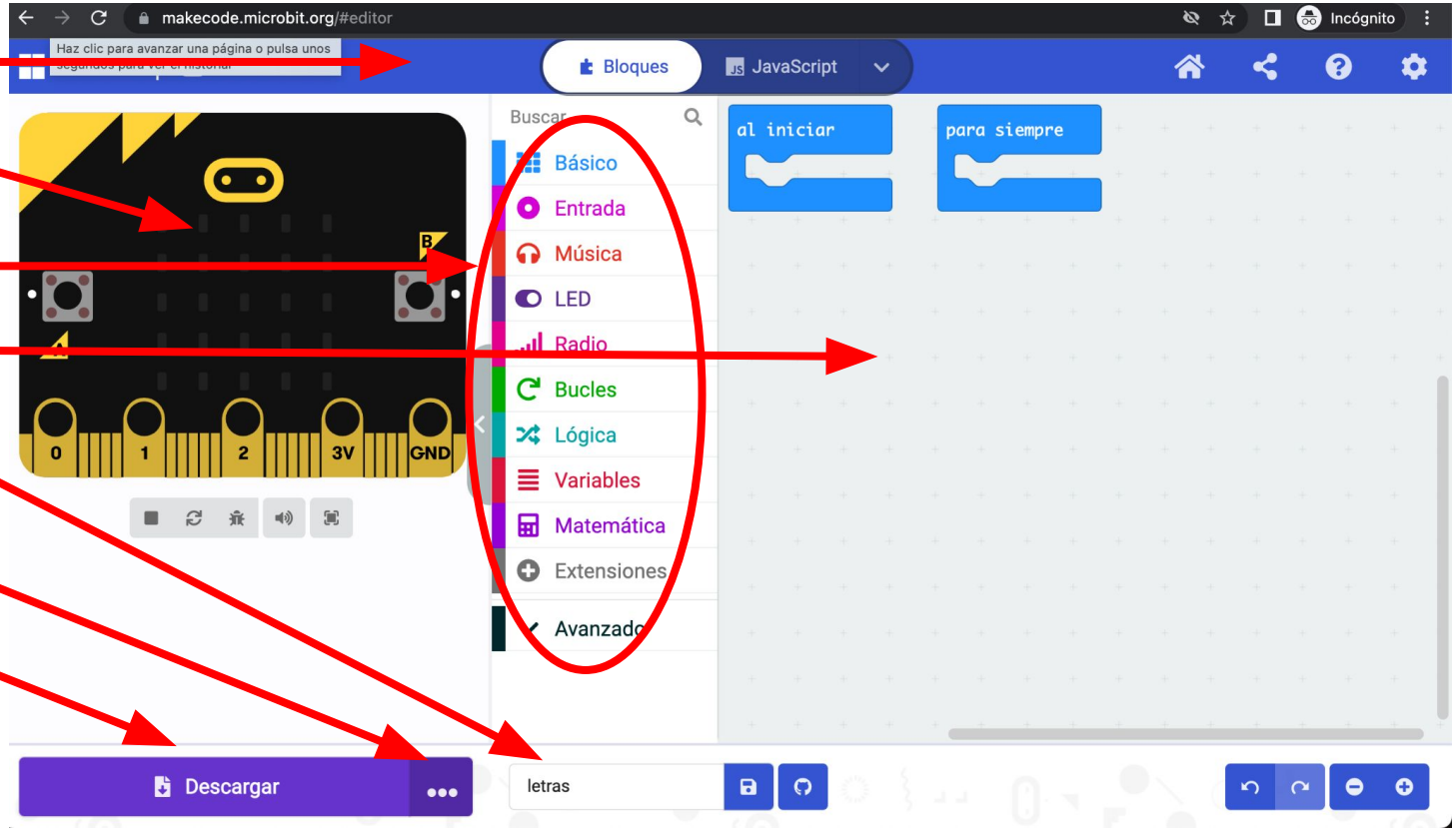
Bloques

Código

Nombre

Conexión

Descarga

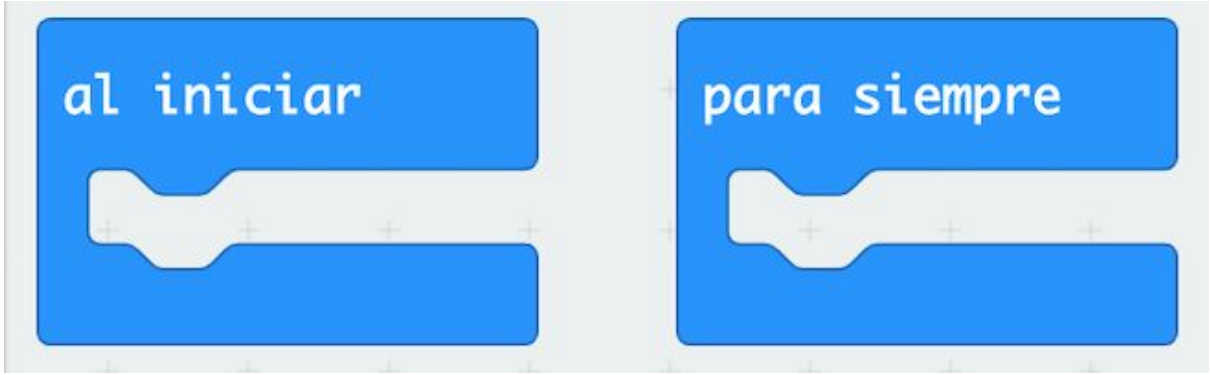


Código por defecto

El código de muestra tiene dos partes:

- Al iniciar - aquí incluiremos el código que se ejecuta una vez
- Para siempre - aquí tenemos el código que se ejecuta repetidamente

El simulador de la izquierda se activará cuando escribamos un programa



¿Cómo dibujar un gato y un ratón en la pantalla?

Para hacer un juego con dos personajes en la pantalla del micro:bit debemos simularlos con dos puntos:

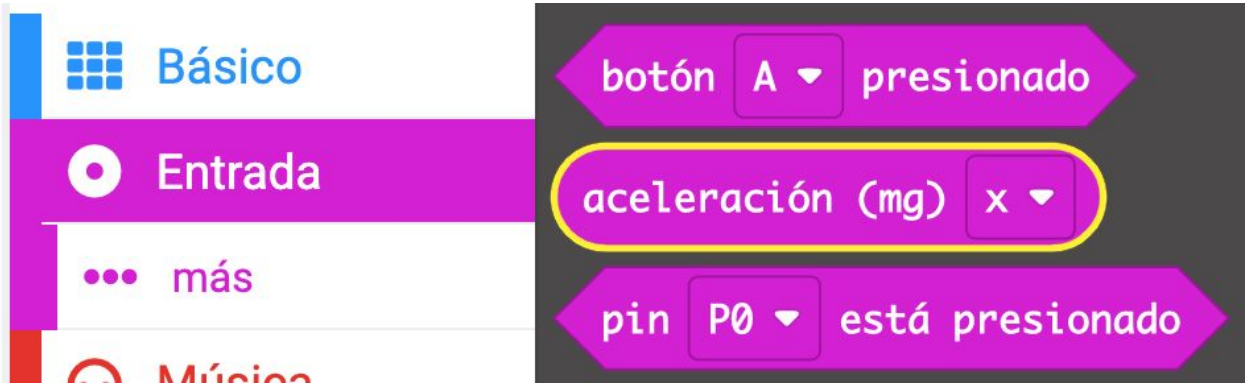
- pondremos un **punto brillante** para el **perseguidor** (gato)
- podremos un **punto más débil** para el **otro personaje** (ratón)

Veremos más adelante cómo crear los personajes

¿Cómo mover los personajes?

La forma más simple de mover un personaje en la pantalla es inclinando la tarjeta **micro:bit** hacia un lado o a otro

Ya sabemos que la tarjeta **micro:bit** tiene un **detector de inclinación** que nos permite conocer hacia dónde se inclina la tarjeta. El valor se llama **aceleración**

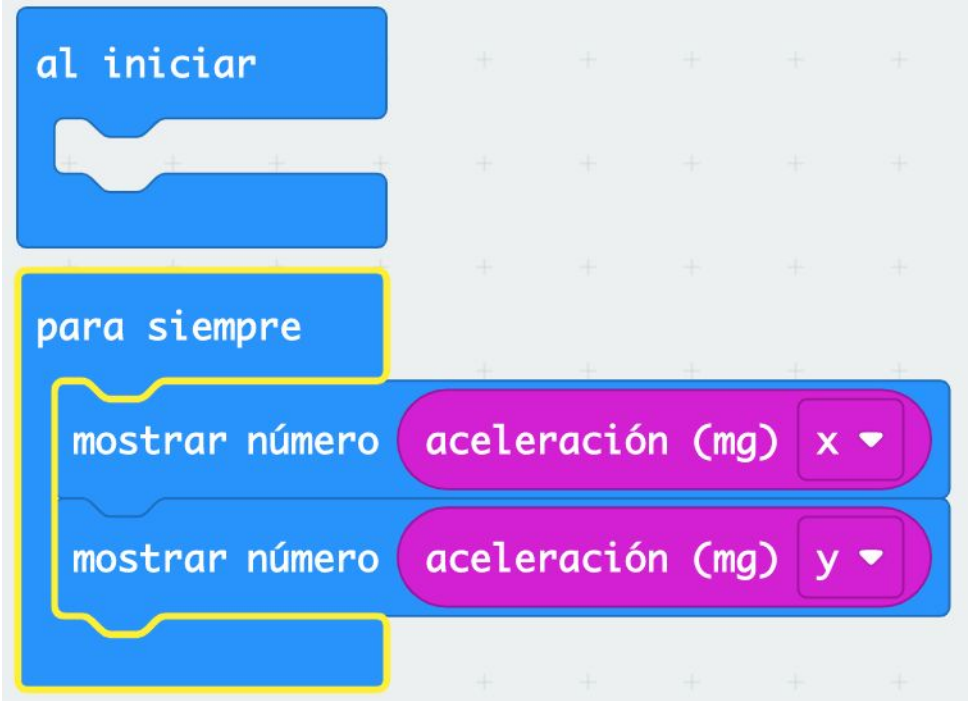


Mostremos la aceleración

Para que el simulador nos permita visualizar la **aceleración** vamos a escribir su valor en la pantalla de **micro:bit.** Dentro de los bloques **Básico** encontramos

y en **Entrada** tenemos la **aceleración**

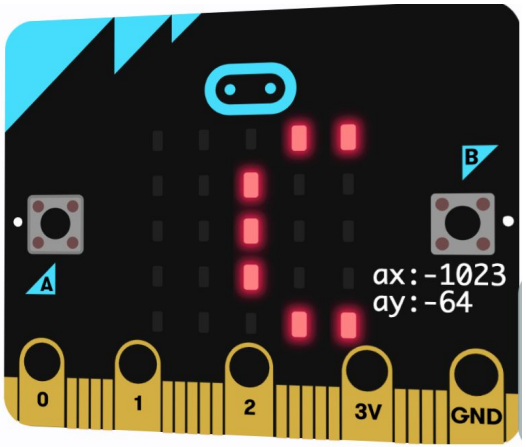
que tiene varios valores. Nos interesa el movimiento en **X** y en **Y**

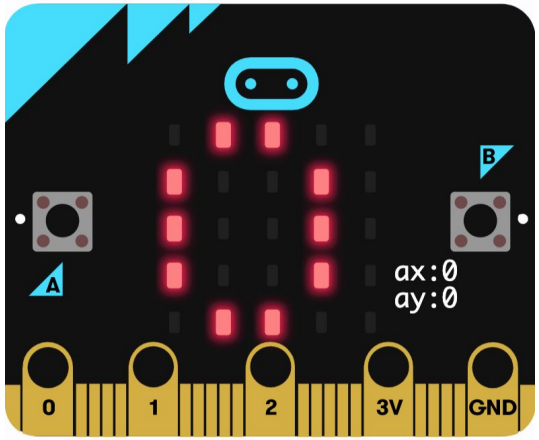


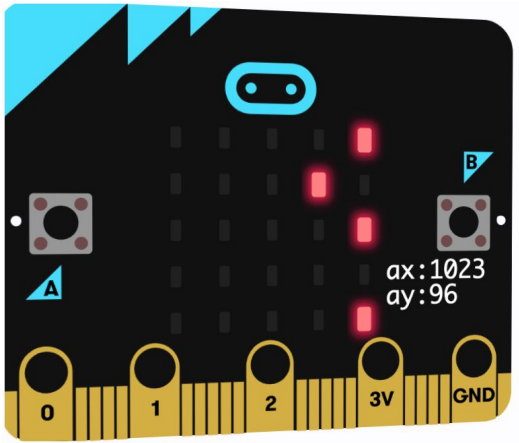
Observemos

¿Qué ocurre en la simulación?

Cuando inclinamos la tarjeta hacia la **izquierda** aparece **en blanco** el valor **ax:-1023** y si la inclinamos hacia la **derecha ax:1023**. En el **centro** tenemos **ax:0**







Valores de aceleración

Los valores de la aceleración varían:

- ax:-1023 inclinada hacia la izquierda
- ax:0 en el centro
- ax:1023 inclinada hacia la derecha

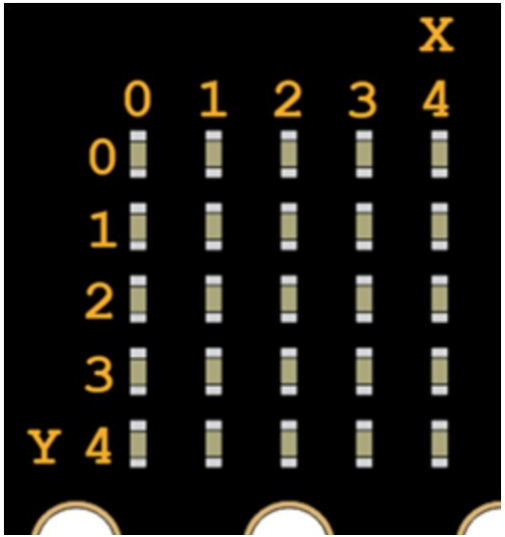
- ay:-1023 inclinada hacia arriba
- ay:0 en el centro
- ay:1023 inclinada hacia abajo

¿Cómo utilizar la aceleración para mover un punto?

Si podemos dibujar un punto en la pantalla, moviendo la tarjeta podremos moverlo hacia un lado y hacia el otro

¿Qué coordenadas tienen los leds?

Los leds de la pantalla tienen las coordenadas que se muestran



¿qué coordenadas tiene el led central?

X:2 Y:2

Dibujar un punto en la pantalla

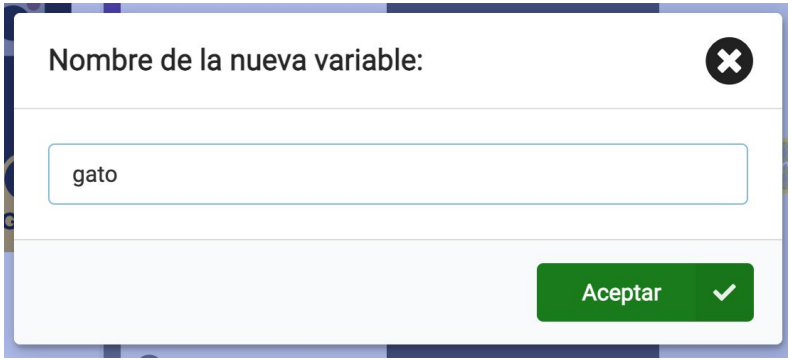
Para dibujar un punto en la pantalla tenemos varias opciones. Vamos a elegir una que nos permite hacer nuestro **juego** muy fácilmente

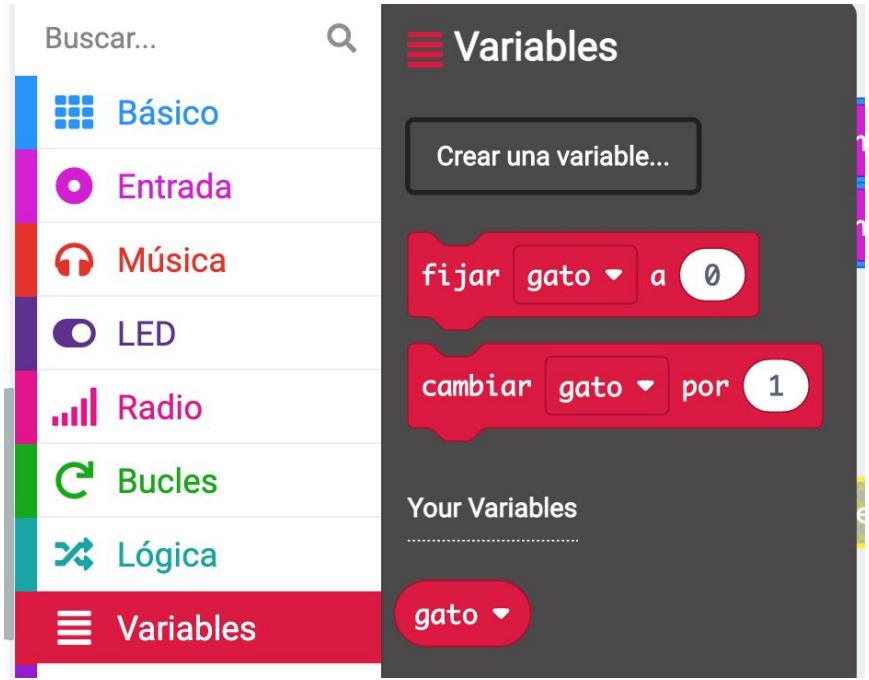
El punto se denomina **sprite**, que se traduce por **duende,** aunque en **informática** define un **grupo de puntos** que pueden formar una imagen

En **micro:bit** equivale a un **led**

Variable gato

Creamos una **variable** para dar un nombre al **sprite**, por ejemplo **gato**

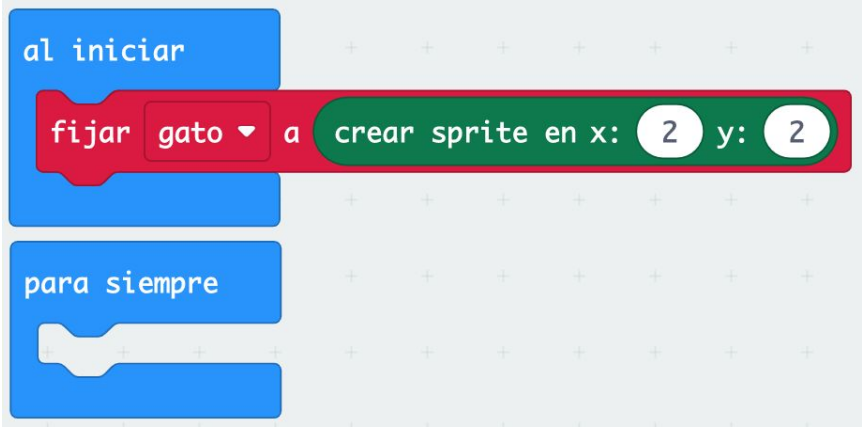




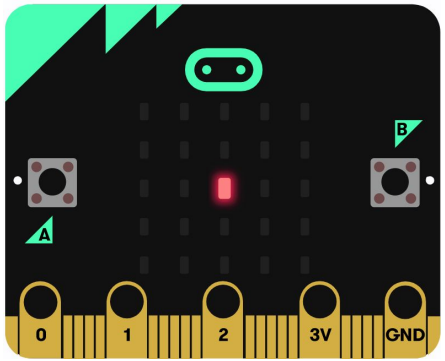
Crear un punto (sprite)

Ahora podemos crear un **punto**, al que llamamos **gato**, con el bloque que encontramos en **Juego** y que permite crear un **sprite** en unas coordenadas que elijamos, en este caso **x:2 y:2**

Cambiamos el programa anterior



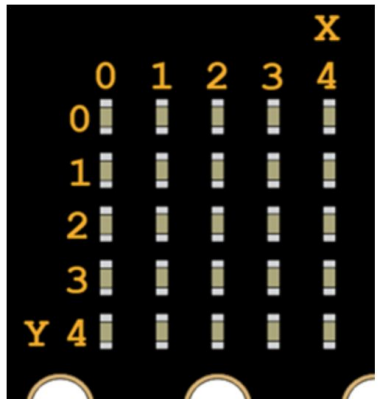
En la simulación debe aparecer un punto en el centro de la pantalla



¿Cómo mover un punto?

Hemos dicho que vamos a mover el punto moviendo la tarjeta

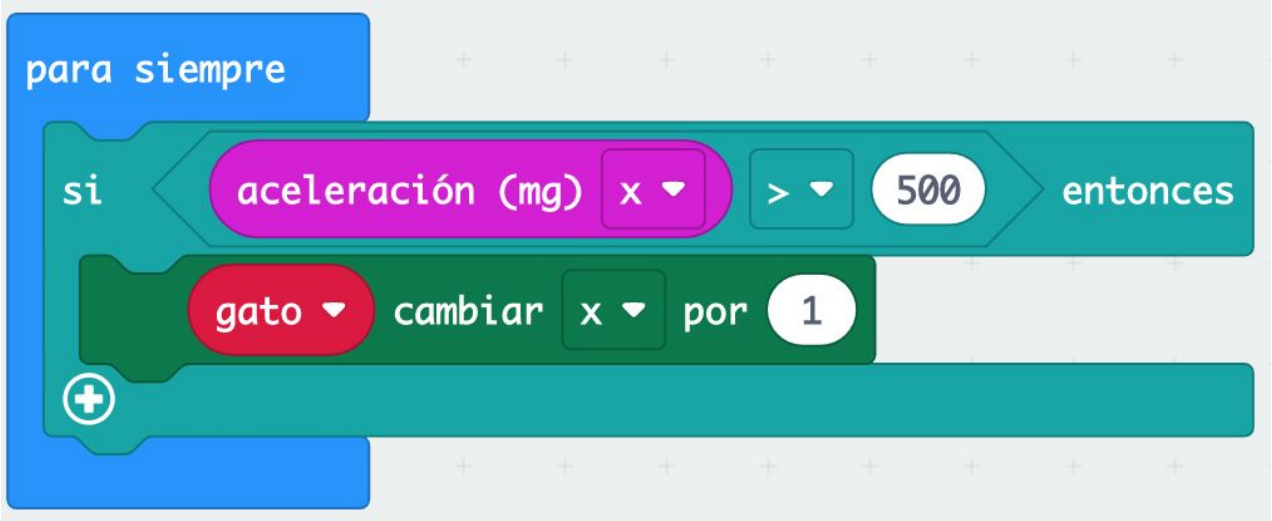
Si movemos la tarjeta hacia la **derecha** hemos de **aumentar** la **coordenada X del punto** (sprite o gato)



En **Juego** encontramos la forma de cambiar las coordenadas **X** o **Y** del sprite

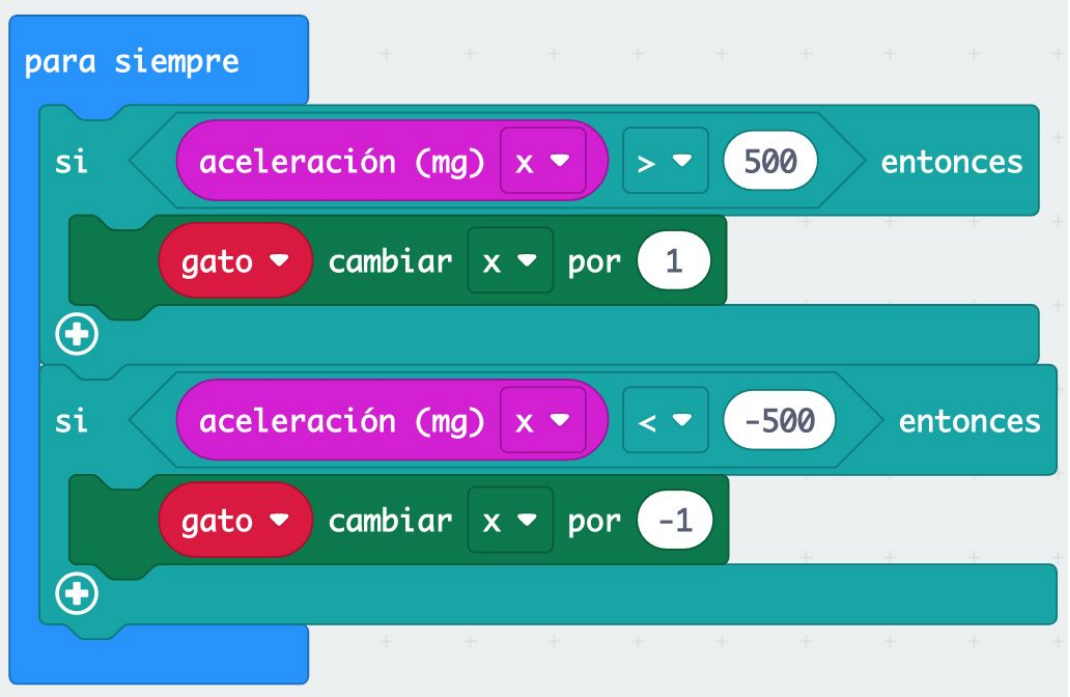
Mover el punto (sprite)

Comparamos la **aceleración X** con un valor, por ejemplo **500**, y si es mayor, que indica que la tarjeta está muy inclinada a la derecha, **aumentamos** la **coordenada X** del gato (sprite) en **1**



Mover el punto (sprite)

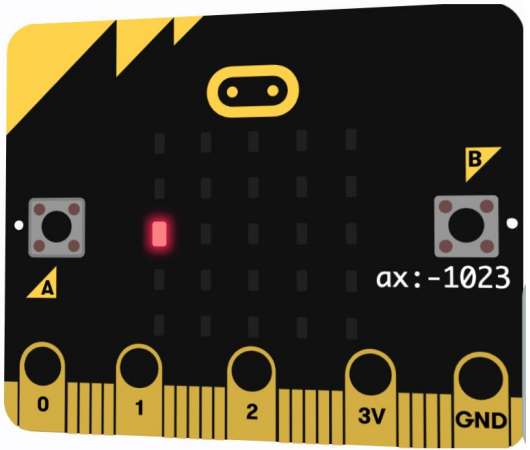
Ahora queremos mover el punto hacia el otro lado. **Comparamos** la **aceleración X** con el valor en negativo, por ejemplo -**500**, y si es menor, que indica que la tarjeta está muy inclinada a la izquierda, **disminuimos** la **coordenada X** del gato (sprite) **(aumentamos en -1)**

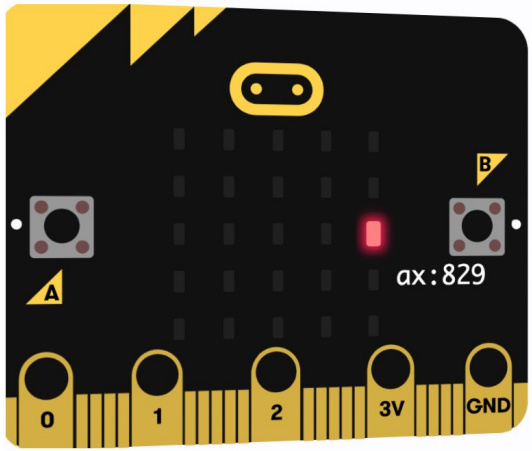


Observemos

¿Qué ocurre en la simulación?

Cuando inclinamos la tarjeta hacia la **izquierda** el punto se mueve a la **izquierda** y cuando la movemos a **derecha** el punto va a la **derecha**





¿Hay límite en las coordenadas?

Hemos visto que las coordenadas de los leds van de 0 a 4

En nuestro programa **no hemos puesto ningún límite** y sin embargo **funciona** !!

El código interno que maneja los **sprite** se encarga de vigilar los **límites de las coordenadas**. De todas formas, es mejor que cambiemos un poco el programa y añadamos el control de los límites. Solo podemos sumar uno si **X < 4**

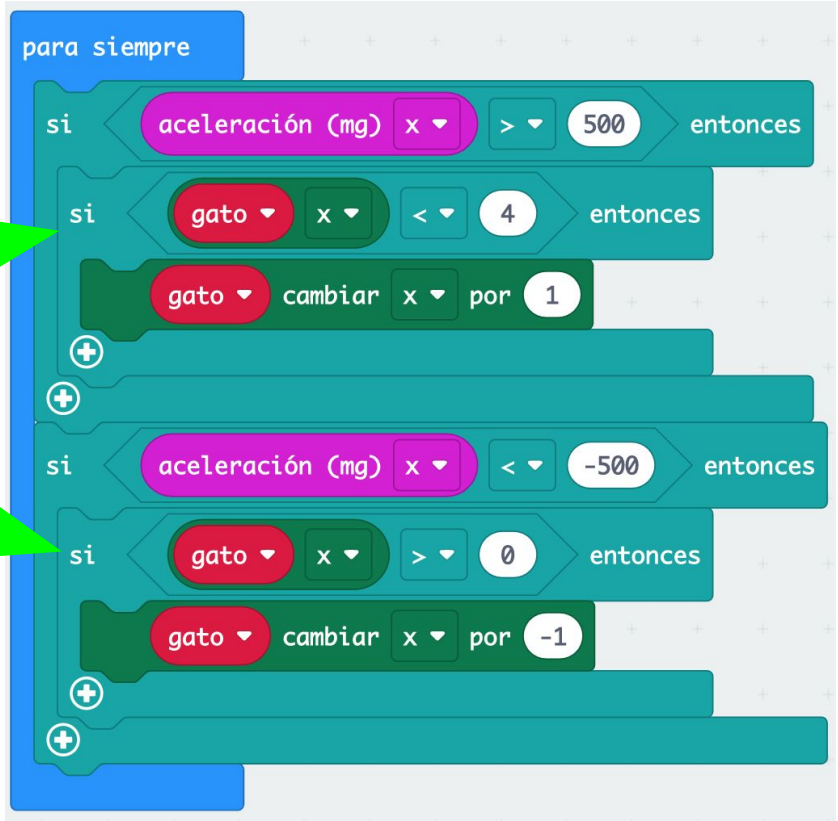
Programa bien escrito

Añadimos las dos comparaciones

si X < 4 podemos aumentar

si X > 0 podemos disminuir

En la simulación no cambia nada !!



¿Nos falta algo?

Si, claro, el punto se mueve de lado pero no se mueve de arriba hacia abajo.

Hemos de copiar los mismos bloques, pero cambiando las X por una Y

Programa mejorado

Ahora ya podemos mover el punto en todas las direcciones

Verifica bien que sean

- seis X
- seis Y
- > 500 < 4 cambiar por 1
- < -500 > 0 cambiar por -1

Observemos

En la simulación, trata de dejar el punto en el centro. No parece posible

¿qué ocurre?

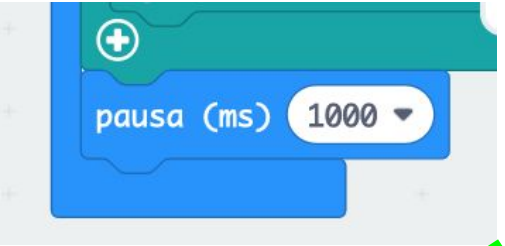
¿qué hace nuestro programa?

Solo mira la aceleración y por tanto no da tiempo a mover la tarjeta

Vamos a añadir un tiempo de espera

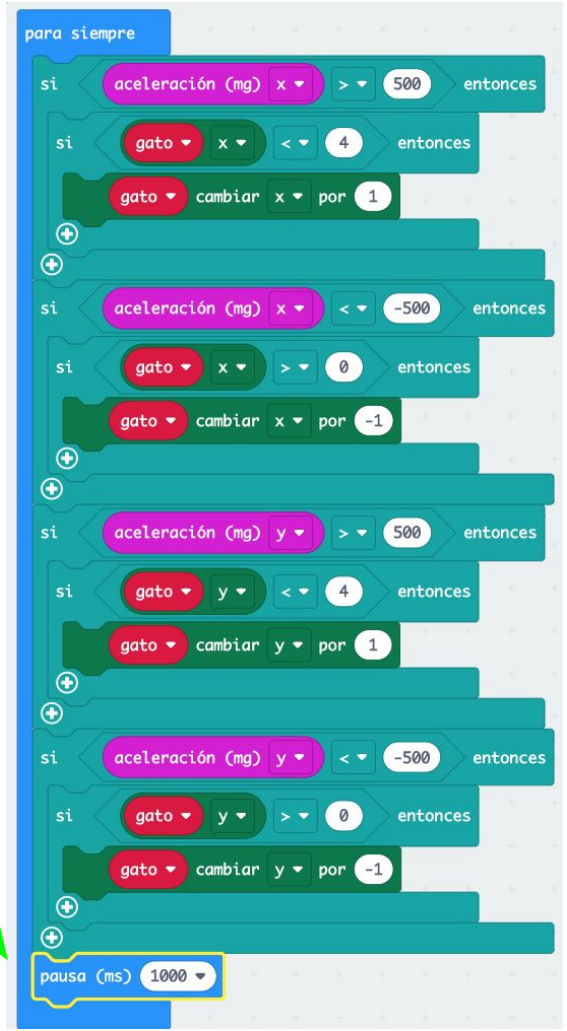
Programa con pausa

Sólo hemos añadido un pausa al final



¿Va mejor el juego?

Si, quizás es un poco lento



¿Podemos mejorarlo?

Podemos **crear una variable** para poner el tiempo de espera, y la podemos llamar **tiempo_de_reaccion**

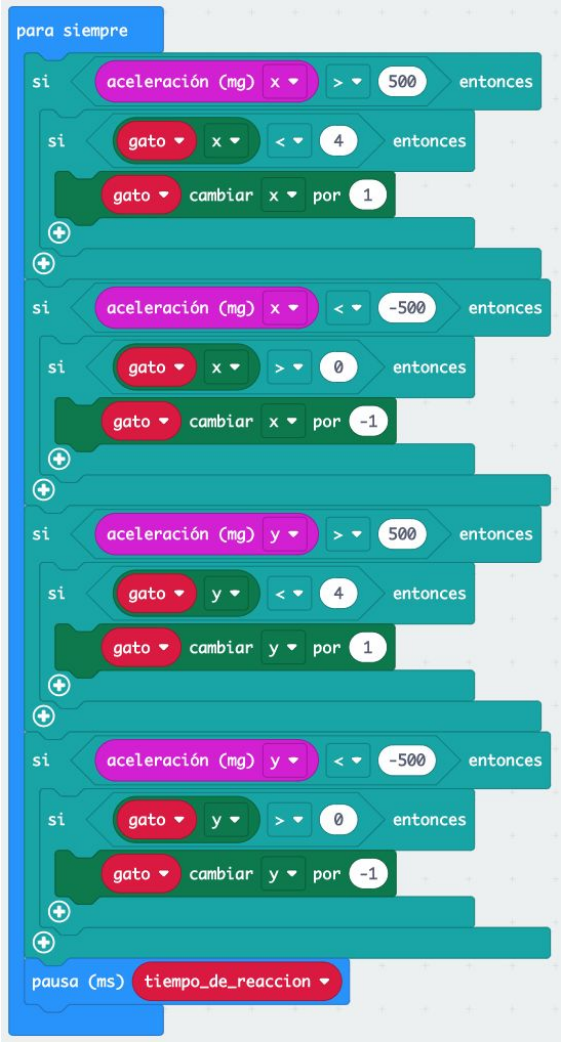
Nos falta **definir** el valor. Lo haremos como siempre en el bloque **al iniciar**

Programa con pausa

Podemos probar con un valor 500

¿Va mejor el juego?

Seguramente ahora va bastante bien



Observemos

En la simulación, ¿somos capaces? de:

- poner el punto en el centro
- poner el punto en cada esquina
- poner el punto en otras coordenadas

Mejor lo llevamos a la tarjeta de **micro:bit** y lo probamos de verdad, para ver que tal tenemos nuestro **equilibrio**

RESUMEN DE EQUILIBRIO

Ya tenemos un programa completo que nos permite **probar el equilibrio** con **micro:bit**

Ahora es el momento de copiarlo a nuestra tarjeta **micro:bit** y probarlo de verdad.

Recuerda los pasos:

1. Conectar el micro:bit al ordenador
2. Emparejar el **micro:bit** (**Connect device**)
3. Descargar el código al **micro:bit**

¿Habíamos hablado de un ratón?

Para **seguir con el juego** nos falta añadir un **ratón** al que perseguir con el movimiento del **gato**

Tal como ya hemos dicho, el **ratón** será otro **punto** (sprite) en la pantalla, y lo diferenciaremos haciendo que tenga **menos luminosidad** que el **gato**

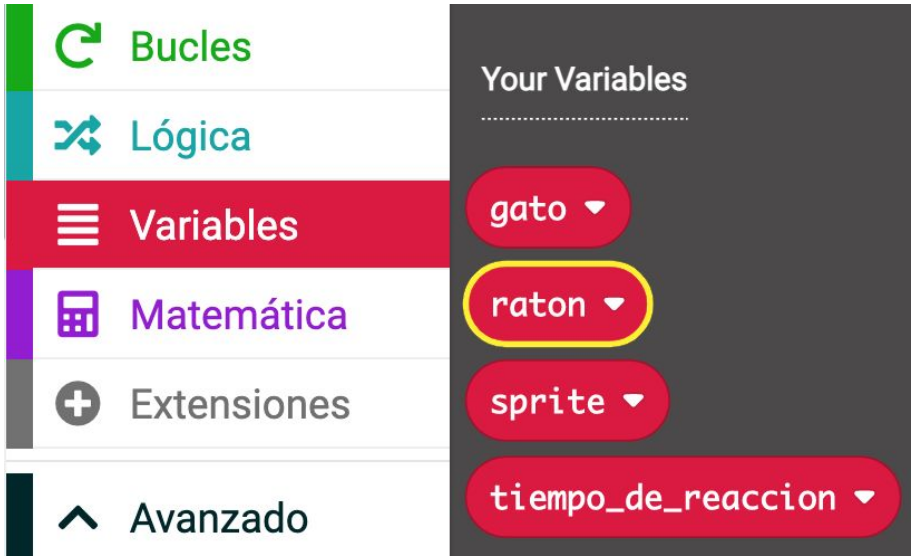
Necesitamos pues:

- crear una variable y un sprite
- que el punto sea menos luminoso
- que se mueva solo por la pantalla

Nuevo personaje

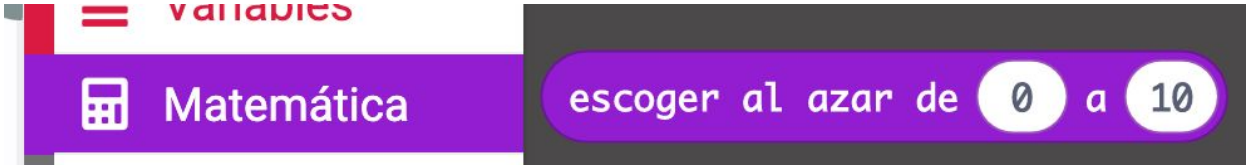
Creamos una nueva variable que llamaremos **raton**

Igual que hemos hecho con el **gato** creamos un sprite para el **raton**, pero necesitamos que aparezca en cualquier sitio



Número aleatorio

Entre los bloques de **Matemática** hay uno que nos da un **número al azar** (**número aleatorio**), entre dos valores que definamos



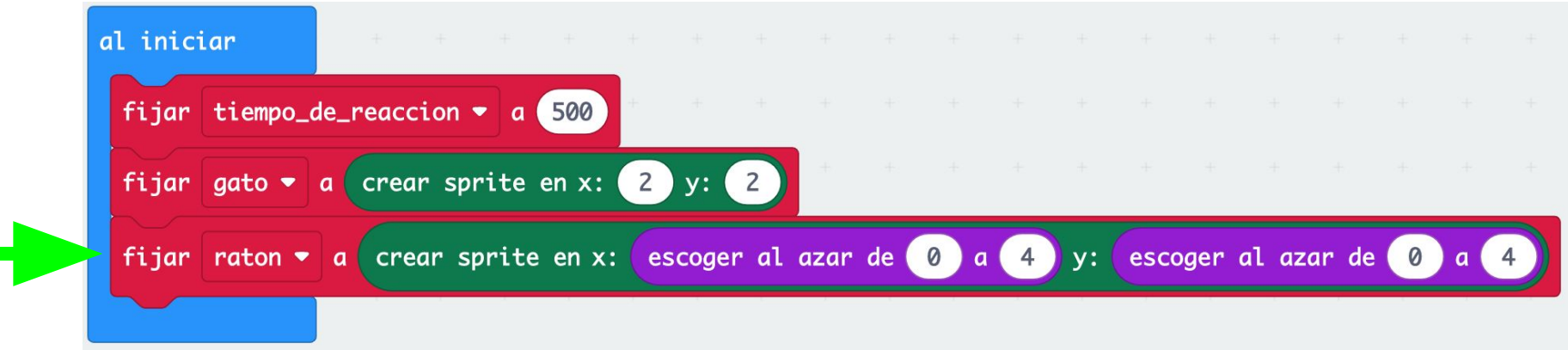
Recordemos que las coordenadas de los led van de 0 a 4, por lo que necesitamos usar estos valores

Posición aleatoria

Creamos el sprite del **raton**, igual que hemos hecho con el **gato**

Cada una de sus **coordenadas X e Y** deben ser aleatorias

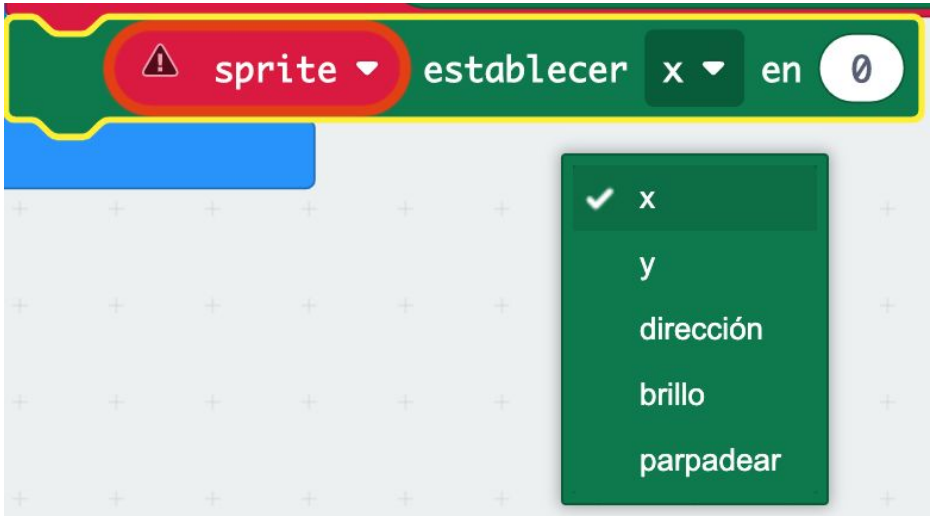
Ponemos **escoger al azar de 0 a 4** para cada coordenada



Cambio de brillo

Entre los bloques de **Juego** encontramos la forma de fijar algunos valores para un punto (sprite)

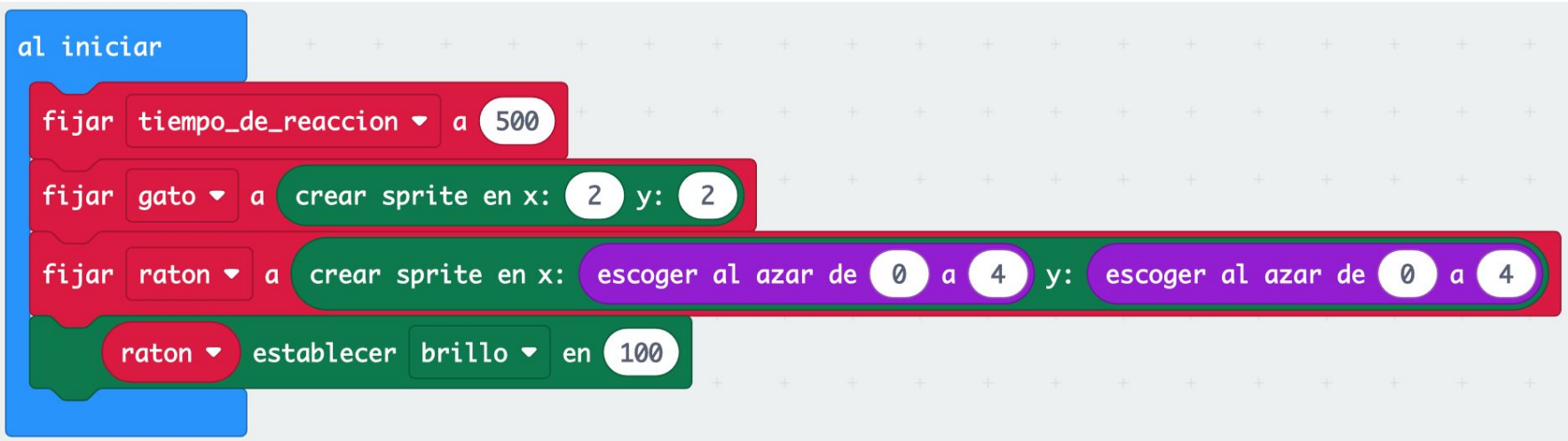
Desplegamos las opciones y vemos que podemos fijar el **brillo**



Ratón creado

El programa completo para crear el **gato** y el **raton** queda muy sencillo

El **gato** se crea en **x:2 y:2** y el **ratón** puede aparecer en **cualquier** sitio
(si, puede darse la mala pata que aparezca en el mismo sitio que el gato)



¿Se tocan el gato y el ratón?

Para **completar el juego** debemos detectar cuando el **gato** ha llegado a la posición del **ratón**

Debemos **comparar las posiciones** de ambos para ver si son iguales. Deben coincidir la X de ambos y la Y de ambos

Si el **gato** está **sobre** el **ratón** debemos:

- aumentar la puntuación del juego
- cambiar de posición el ratón

¿Cuando se tocan el gato y el ratón?

¿Cómo comparamos las posiciones?

Afortunadamente, entre las opciones de **Juego** encontramos una que nos facilita mucho el trabajo

que deberemos modificar para que diga

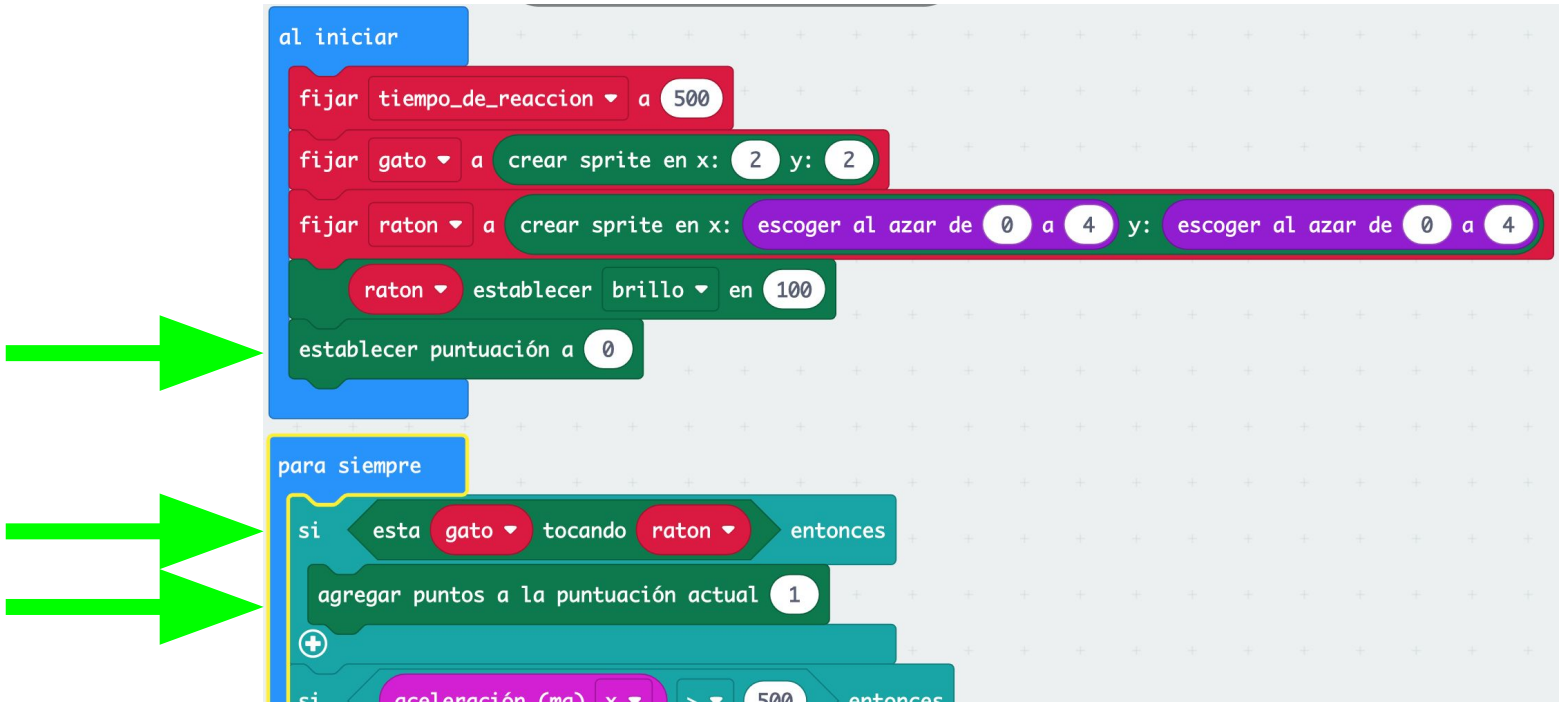
Puntuación del juego (1)

Los bloques de **Juego** nos facilitan el trabajo de llevar la cuenta de puntos

El primer bloque pone a cero el contador y el segundo aumenta la **puntuación**

Puntuación del juego (2)

Añadimos al programa los bloques de puntuación

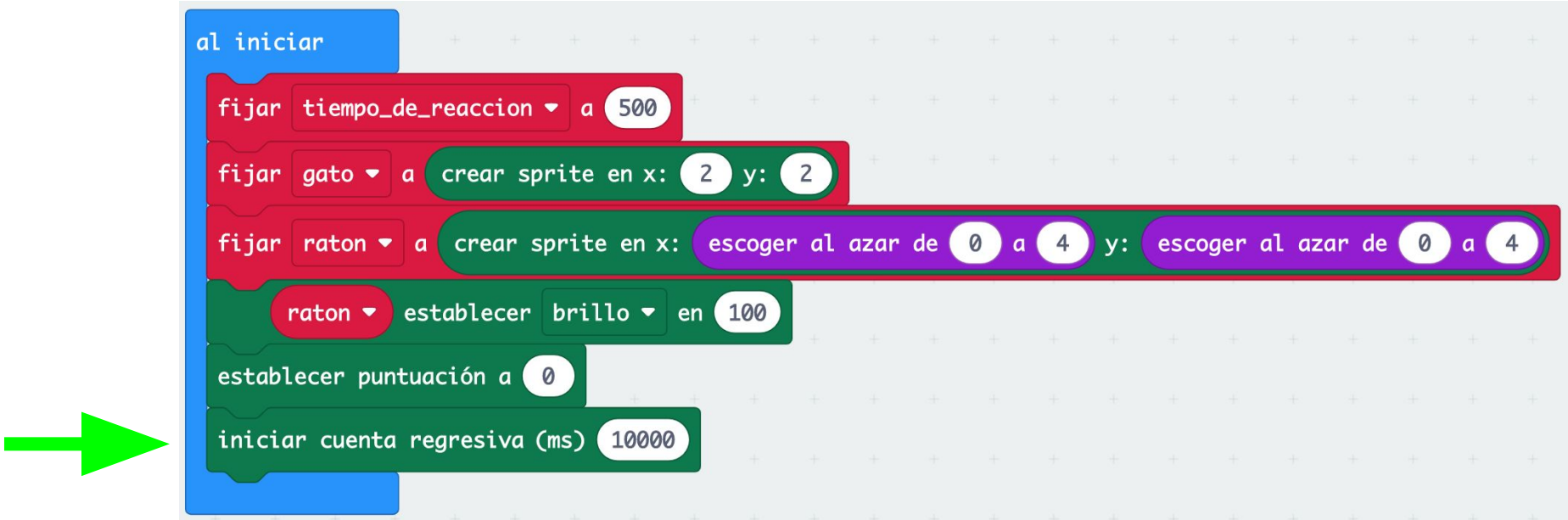


Tiempo de juego (1)

Los bloques de **Juego** permiten controlar el tiempo que dura el juego.
Iniciar cuenta regresiva permite fijar el tiempo que durará el juego

Tiempo de juego (2)

Añadimos al programa el control del tiempo que durará el juego (10 segundos)



Observemos

En la simulación veremos los dos puntos, el **gato** en el centro y el **ratón** en otro sitio

Si **movemos** la tarjeta el **gato** se mueve y si lo ponemos **sobre el ratón** se encienden todos los leds para indicar que **hemos ganado un punto**

El juego continúa y al llegar a 10 segundos se acaba

En pantalla vemos **“Game Over. Score: 1”**

¿Falta algo?

Bueno, si, el juego parece muy corto. Es fácil de arreglar, basta con **aumentar el tiempo de juego**

Si hemos tenido tiempo de fijarnos en lo que ocurre tras lograr un punto, vemos que quizás el gato se ha movido, pero el **ratón sigue en el mismo punto**

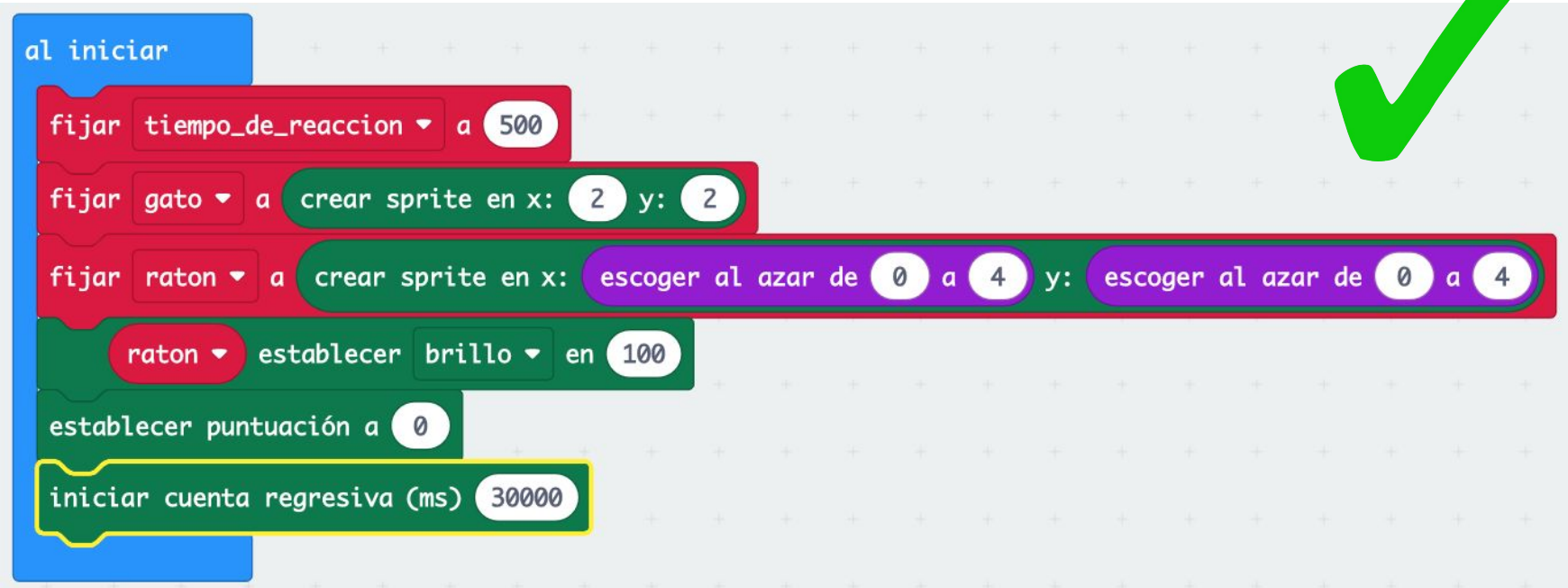
Falta mover el ratón a otra posición. Lo haremos justo después de aumentar la puntuación, cambiando los valores de X e Y

Cambio de posición del ratón

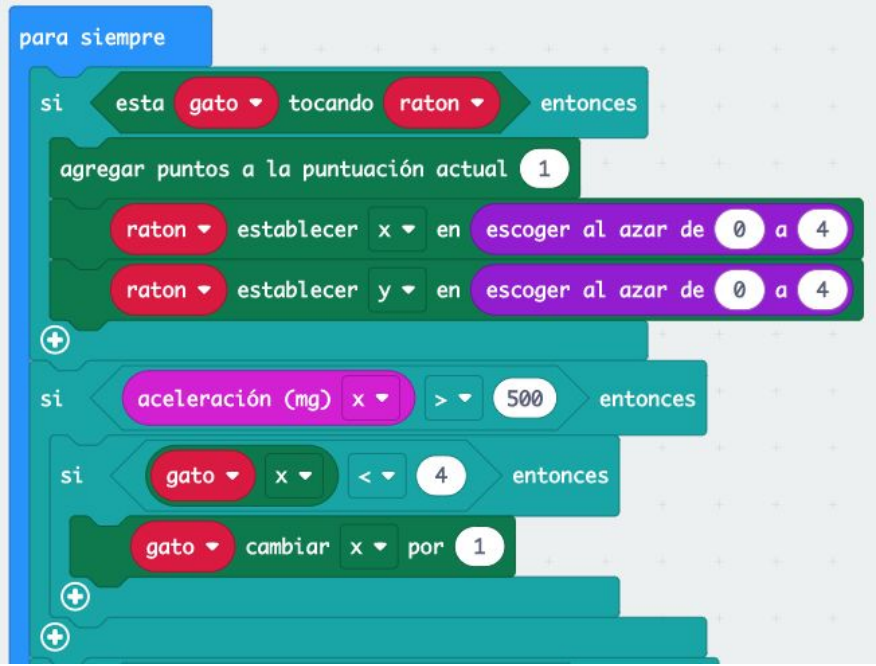
Añadimos dos bloques para cambiar la posición **X e Y** del **raton**

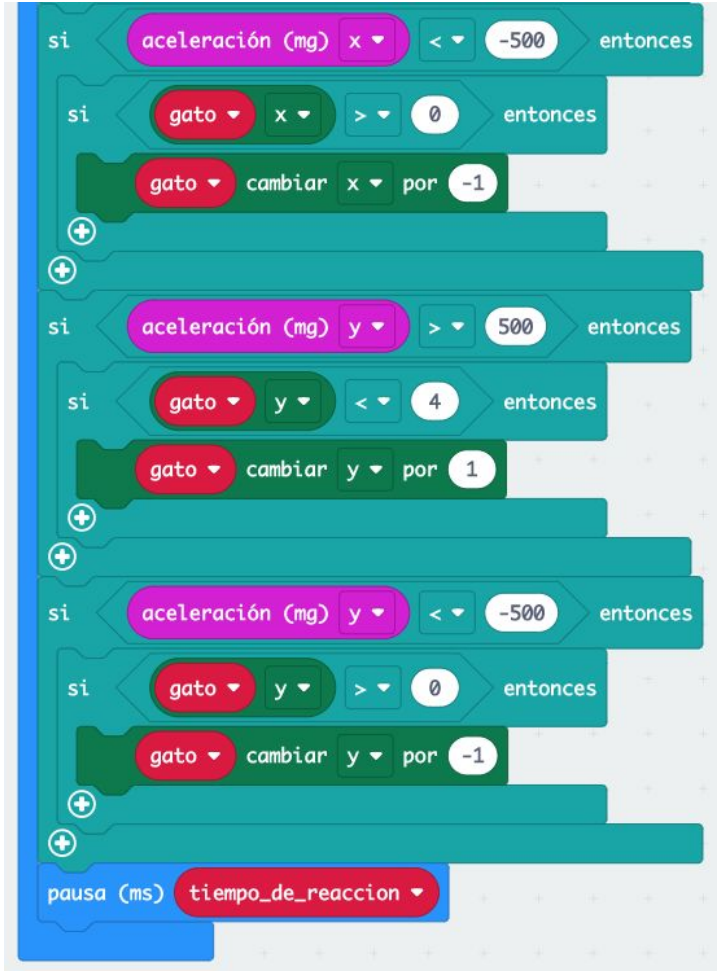
Programa completo (1)

Hemos cambiado el tiempo de juego a 30 segundos



Programa completo (2)





RESUMEN

Ya tenemos un programa completo que nos permite **jugar al ratón y al gato** con **micro:bit**

Ahora es el momento de copiarlo a nuestra tarjeta **micro:bit** y probarlo de verdad.

Recuerda los pasos:

1. Conectar el micro:bit al ordenador
2. Emparejar el **micro:bit** (**Connect device**)
3. Descargar el código al **micro:bit**

Agraïments i Contribucions i Llicència

Han contribuït a la creació d'aquest material

- Joaquin Jimenez Godoy
- Tony Barbosa
- Wouter Molevelt
- Maria Teresa Miras
- Eusebi Calonge

https://steam4all.eu