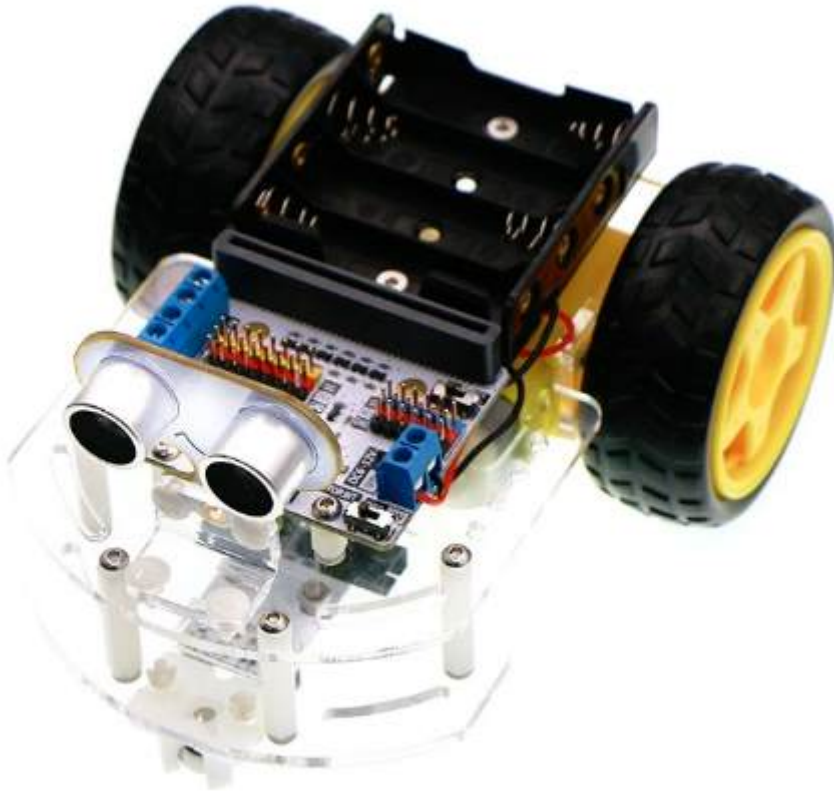


# SmartCar. Dentro del curso

## Microbit+SmartCar

- [SmartCar](#)
- [Reto 1 Musica](#)
- [Sensor distancia](#)
- [Sigue-lineas](#)
- [Reto 5 Mando a distancia](#)

# SmartCar

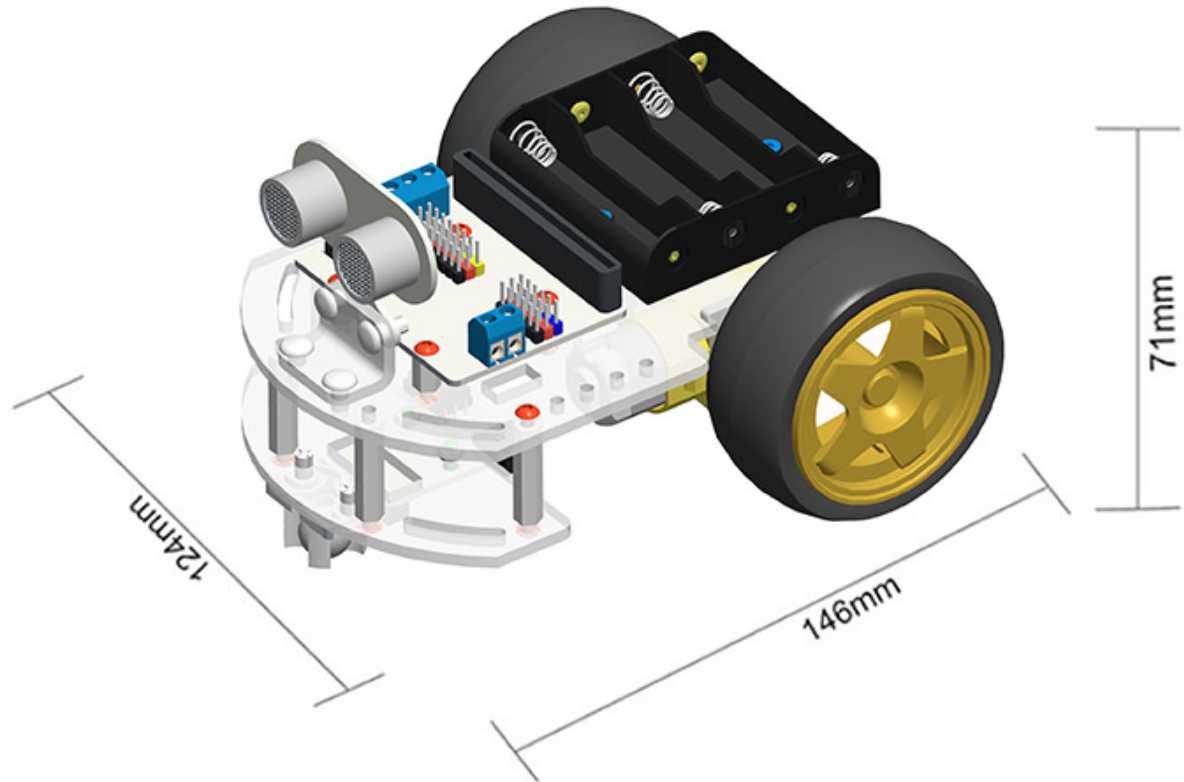


Micro:BIT se queda un poco triste si no se "mueve", pero hay que tener en cuenta que en la robótica el movimiento se paga:

- Adición de motores, chasis, ruedas...
- Necesitas una electrónica añadida de potencia y control de los motores.
- Necesitas unos sensores de lo contrario el movimiento se queda insípido sin reacción al mundo exterior.

La placa de micro:BIT no incorpora movimiento, por eso cuesta tan poco, pero si quieres este extra tienes que pagar ([casi el doble de lo que cuesta micro:BIT](#)). Aún así el conjunto sale más barato que [mBot](#).

Este kit viene con sensor de ultrasonidos, sensor de sigue-lineas y la placa **motorbit** que se inserta la micro:BIT además de tener altavoz y salidas a puertos por si se quiere añadir servos por ejemplo. Para más información de esta placa consultar [esta web](#).

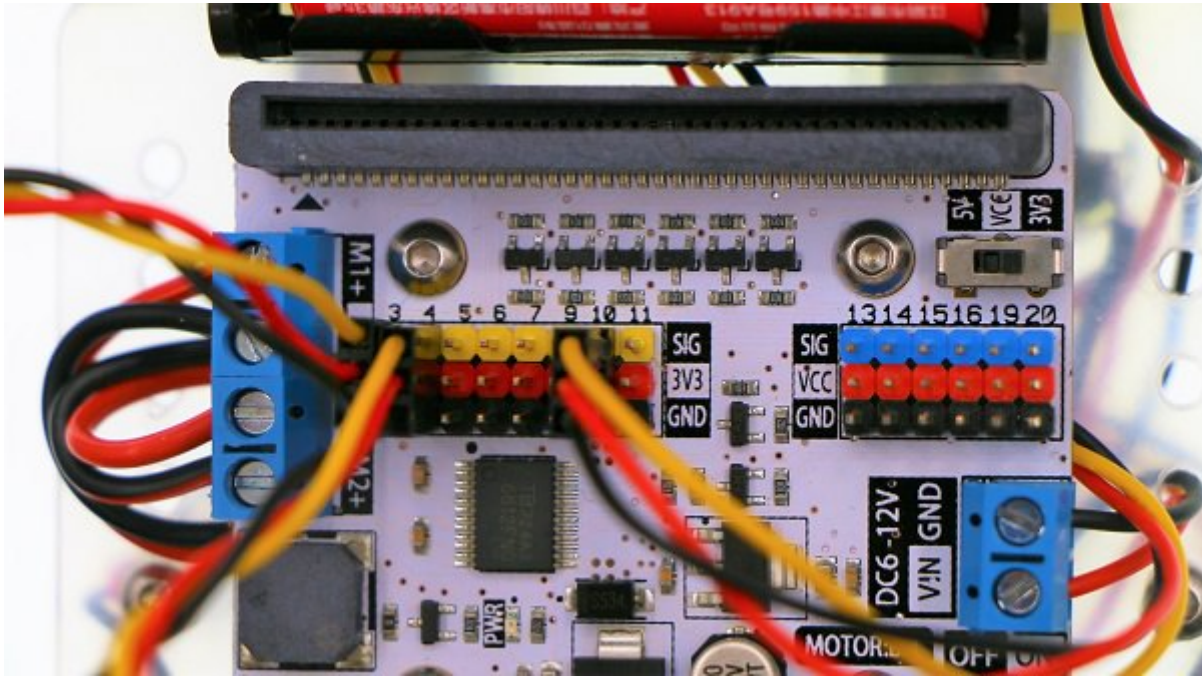


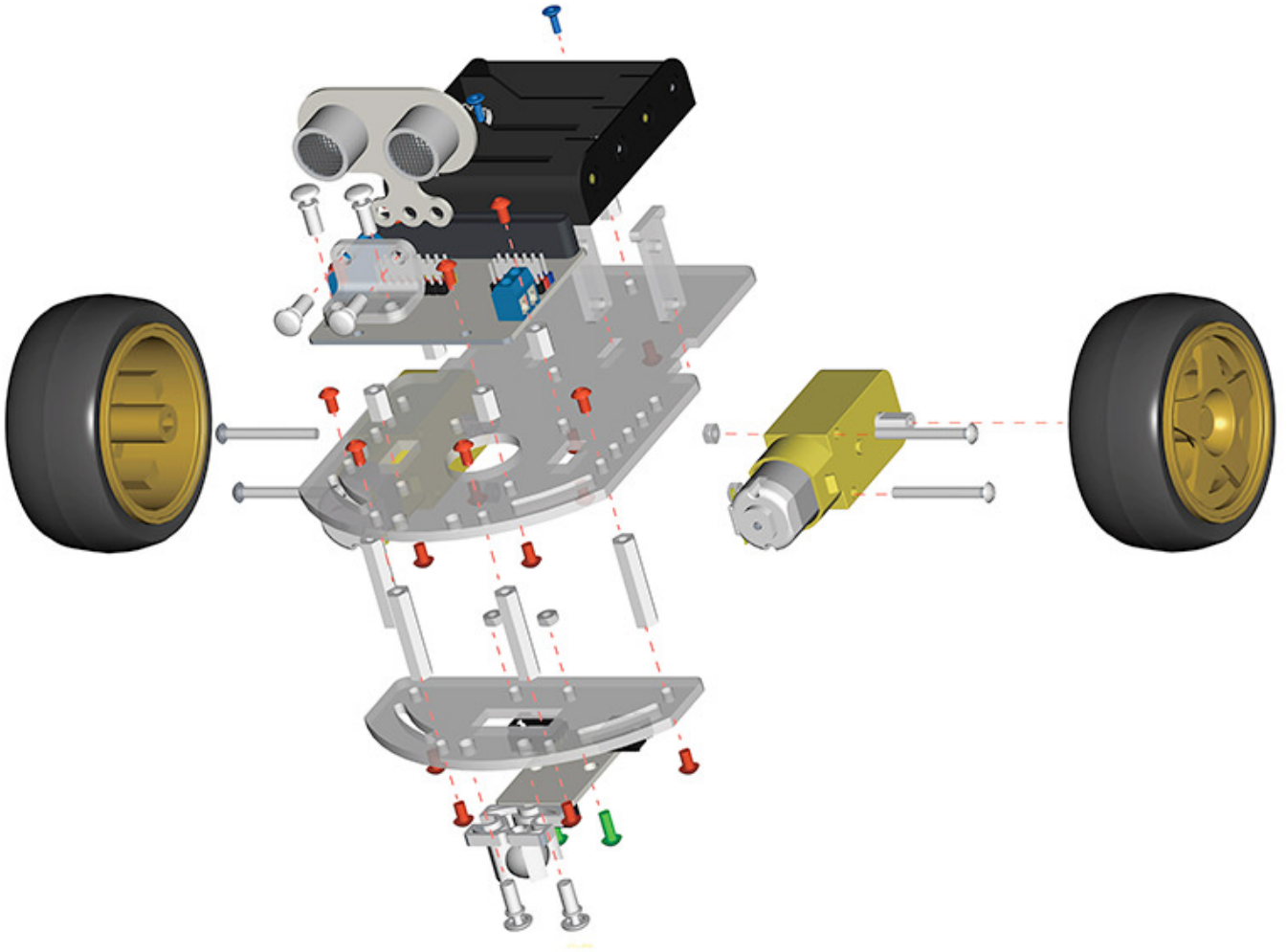
## Montaje

El paquete viene con unas piezas listas para montar que no requieren especial destreza DYC, lo único "difícil" es que hay que **soldar** los cables a los motores (4 pequeñas soldaduras).

El sensor sigue-lineas y el sensor ultrasonidos se puede poner en cualquier pin, pero en este curso lo fijaremos en:

- Sensor ultrasonidos en el PIN 10
- Sensor siguelíneas izquierda en el PIN3
- Sensor siguelíneas derecha en el PIN4

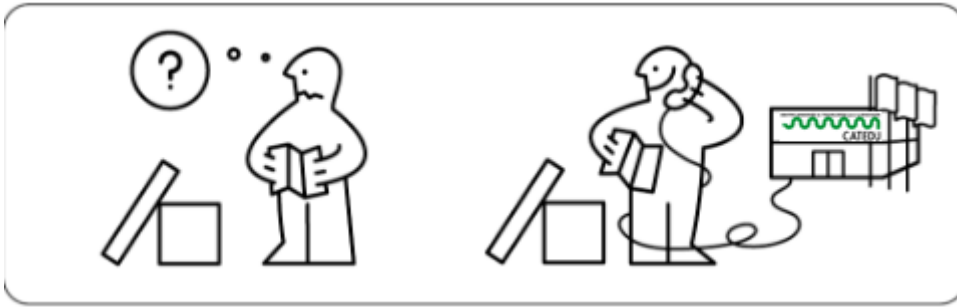




Con un poco de paciencia se hace muy bien

<https://www.youtube.com/embed/EXX9h0i1WU>

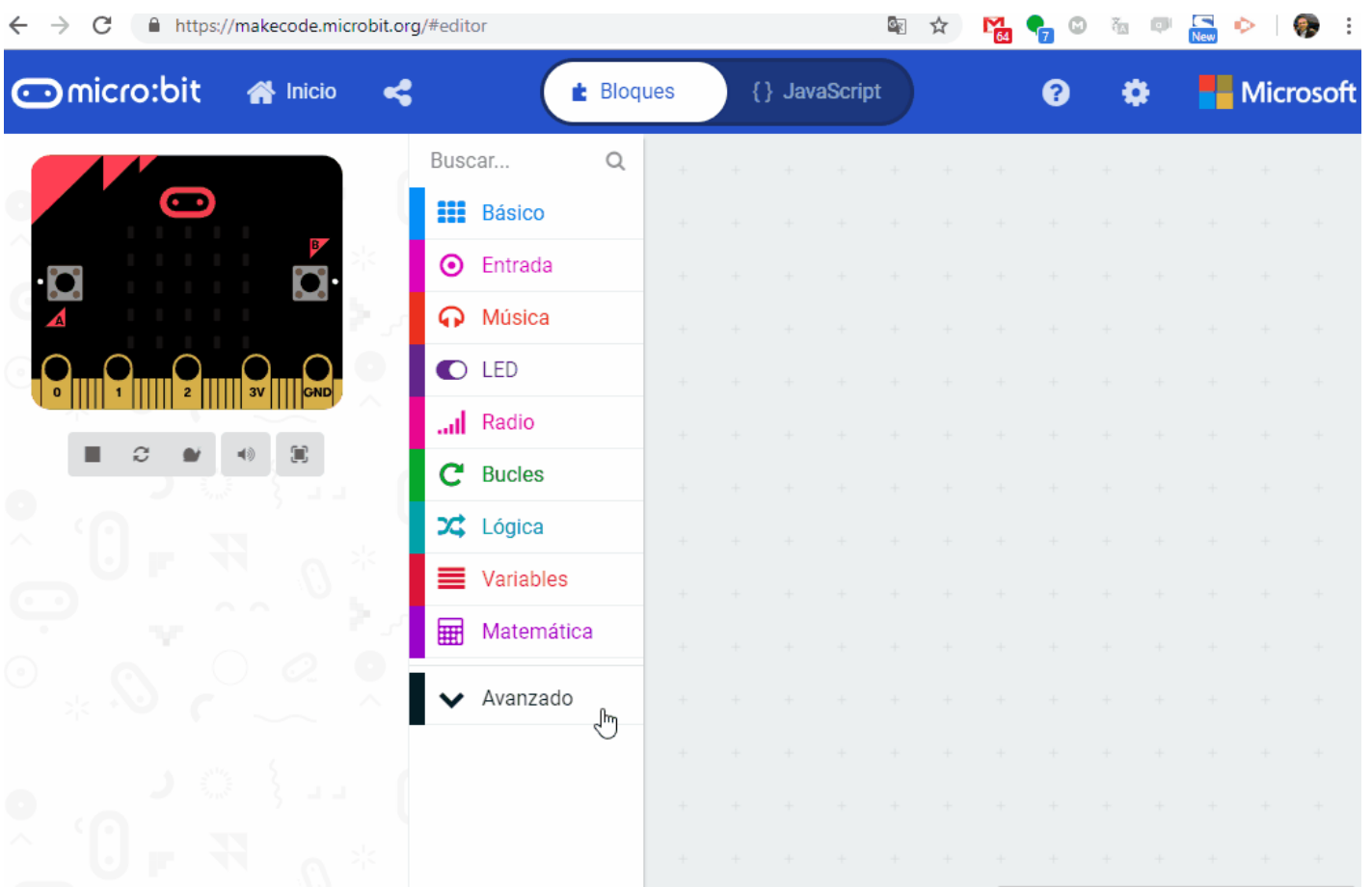
Nosotros no somos comerciales, luego ésto mejor que no :



## Conexión con Makecode

Para tener las instrucciones específicas, hay que entrar en Makecode

<https://makecode.microbit.org/#editor> entrar en **AVANZADO** en **EXTENSIONES** y buscar la extensión **MOTORBIT** instalar, y ya está tenemos instalado las funciones específicas de este coche:



# Reto 1 Musica

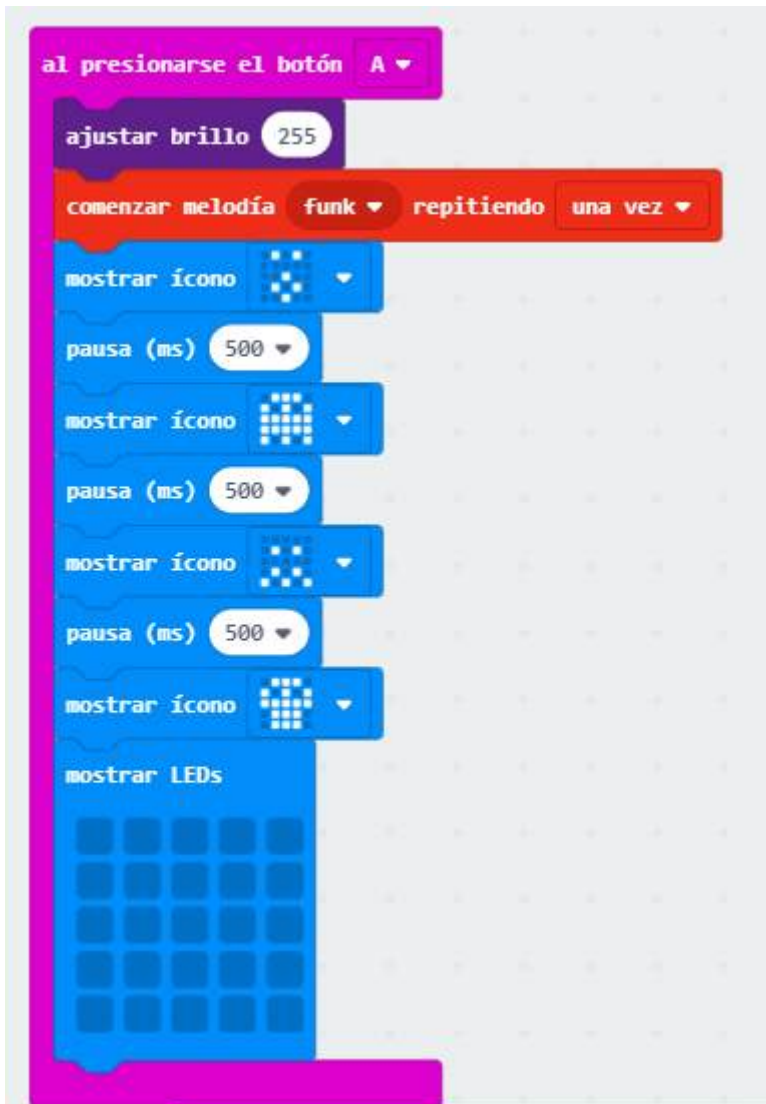
SmartCar tiene incorporado un altavoz ya en el puerto P0 por lo tanto no es necesario hacer cocodrilos o papeles de aluminio [como hemos visto anteriormente](#).

Vamos a hacer un programa que al pulsar el botón A suene música etc...

<https://www.youtube.com/embed/IEuQvhxcw7w>

## Descripción del programa

Es simplemente que si presiono el botón A pues que suene y salgan iconos por la pantalla:



Puedes probarlo en este simulador



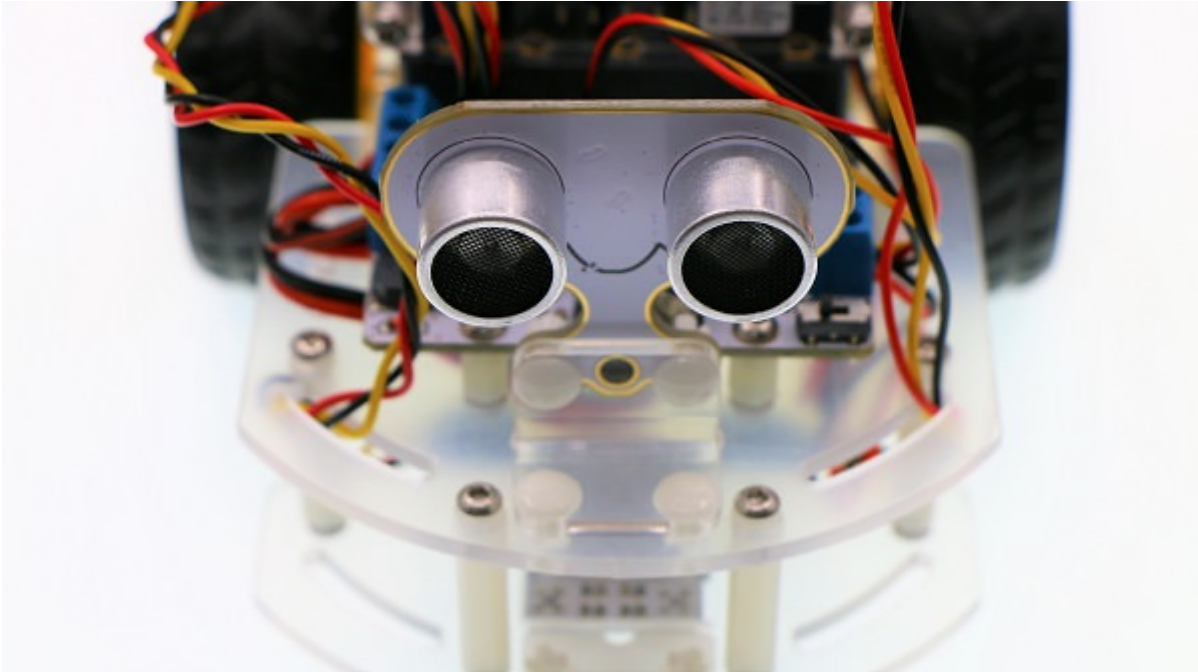
[https://makecode.microbit.org/---run?id=\\_hMmRRea9s76w](https://makecode.microbit.org/---run?id=_hMmRRea9s76w)

Y [aquí](#) lo tienes para descargar:



[https://makecode.microbit.org/#pub:\\_hMmRRea9s76w](https://makecode.microbit.org/#pub:_hMmRRea9s76w)

# Sensor distancia



Al poner la extensión micromotor se añade la instrucción de sonar que está en otro apartado:



Este sensor, en nuestro kit lo conectaremos en **PIN 10** aunque puede estar conectado en cualquier otro. Además recomendamos crear una variable (por ejemplo "distancia") y poner las unidades de medida en cm que son más intuitivas. Esta instrucción se pone al principio del bucle y sólo hay que utilizar la variable **distancia**:



El sensor de distancia funciona igual que los sónares: Uno "ojo" es realmente un altavóz que emite un sonido con frecuencia alta (*ultrasonido, por eso no lo oímos*) y el otro "ojo" es un micrófono que recibe ese pulso. El circuito electrónico calcula la distancia con la diferencia de tiempo en emisión y la recepción del eco, igual que los radares, sónares...

<https://giphy.com/embed/4xGCaTMCO59le>

[via GIPHY](#)

O incluso los murciélagos, delfines...

<https://giphy.com/embed/3o7TKu5aIDY4tU3SXm>

[via GIPHY](#)

## Reto 2 Me quedo a 5cm

El siguiente reto es que el microCar se quede a 5cm

ATENCIÓN vamos a poner el sensor de ultasonidos en el PIN 10

<https://www.youtube.com/embed/iCLnAHLlOlc>

## Solución

La solución se encuentra en esta página [https://www.electfreaks.com/learn-en/motor\\_bit\\_smart\\_car\\_case\\_02/](https://www.electfreaks.com/learn-en/motor_bit_smart_car_case_02/)

El proyecto te lo puedes descargar [aquí](#)

[https://makecode.microbit.org/#pub:\\_J0HabpedtUvY](https://makecode.microbit.org/#pub:_J0HabpedtUvY)

## Reto

Que empiece parado y que se aleje si ponemos la mano, o sea, [intenta cogermelo](#)

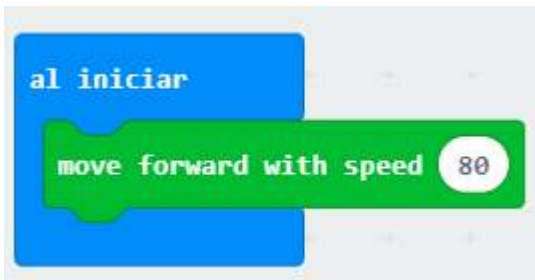
## Reto 3 Evitar obstáculos

¿Cómo no? lo pide a gritos ! vamos a hacer un [roomba](#)

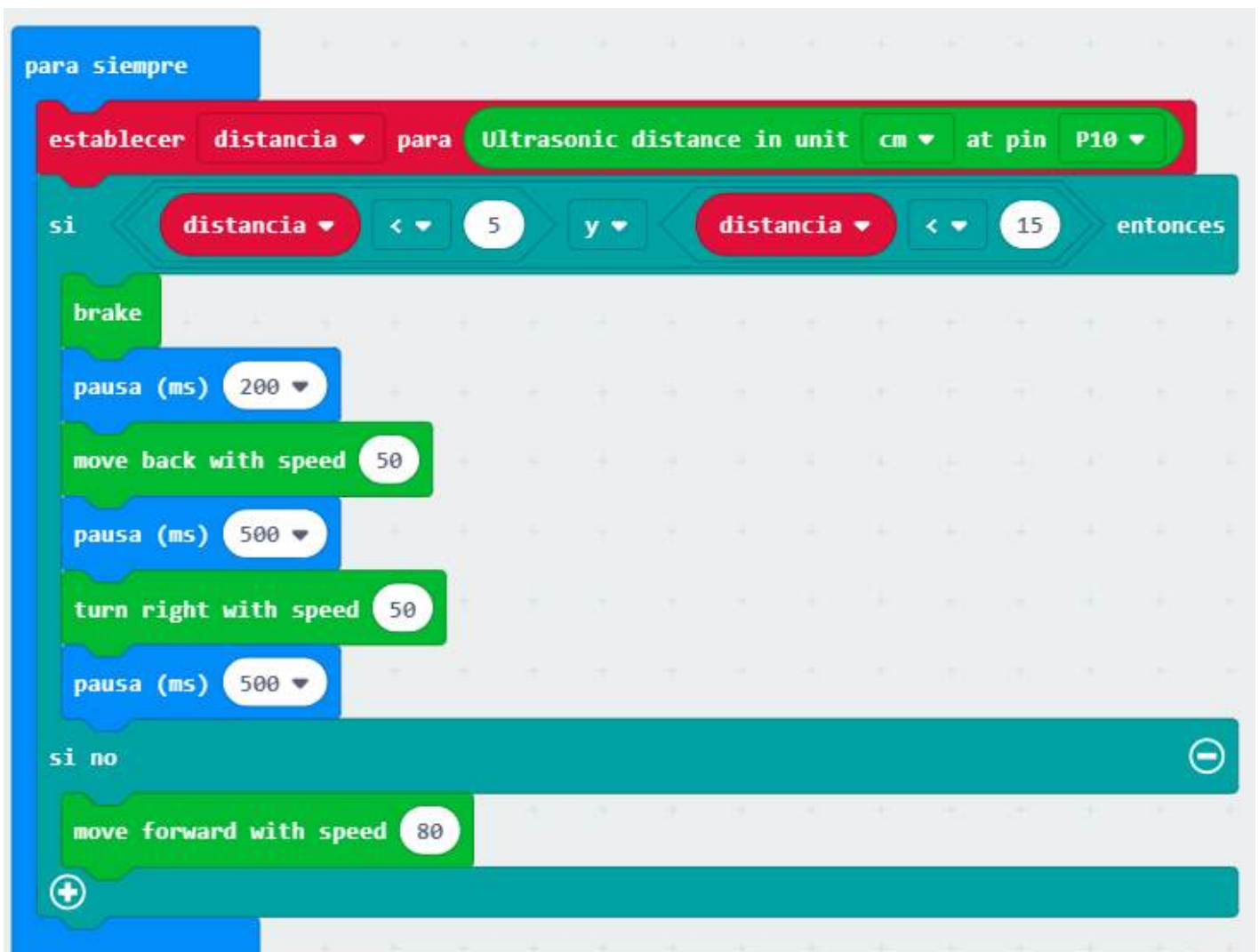
<https://www.youtube.com/embed/nVwOLRhooOc>

## Descripción del proyecto

Empezamos el programa que al iniciar vaya recto:



Luego hacemos el bucle "para siempre": \* El sensor no funciona muy fino, hay veces que da 0 falsos por lo tanto el bucle **si** ponemos una condición **si es mayor de 5 y es menor de 10** para quitarnos estos falsos positivos de obstáculos. \* si encuentra obstáculo, que pare un poco, que *recule* y que gire \* cada instrucción anterior con una pequeña pausa, cuanto más grande sea la pausa más *regulará*, girará etc.. \* si no encuentra obstáculo que siga recto



Aquí lo tienes en editor

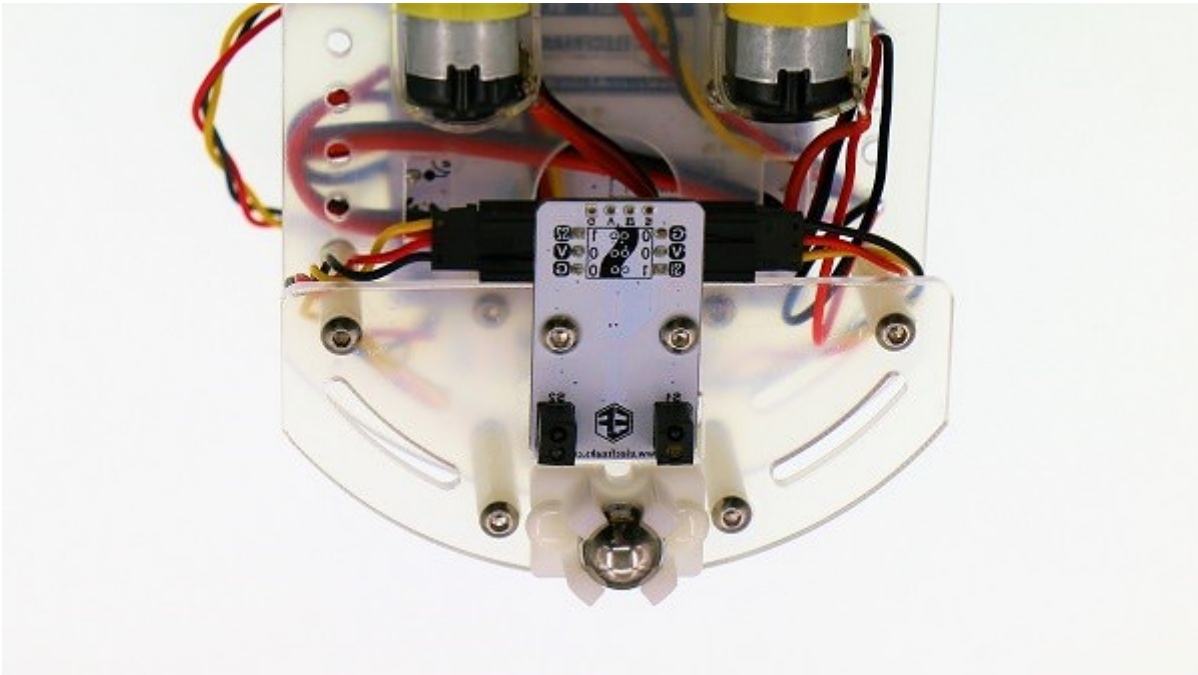
[https://makecode.microbit.org/#pub:\\_5RPLY4TfKXae](https://makecode.microbit.org/#pub:_5RPLY4TfKXae)

## Reto

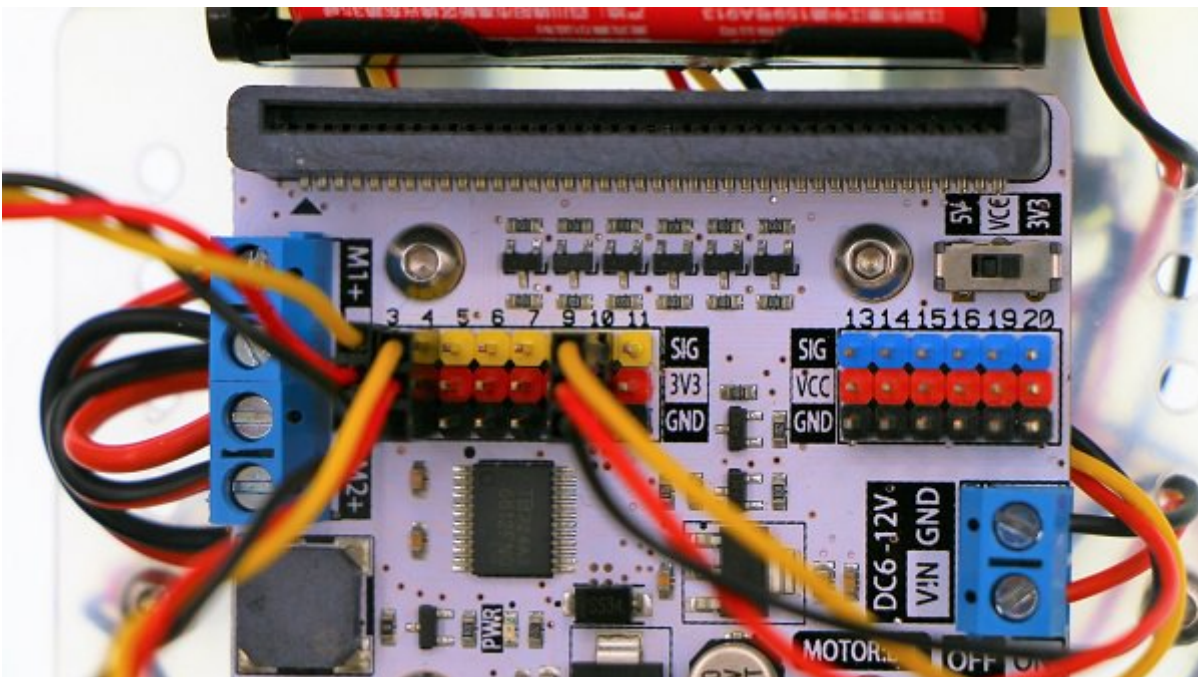
Si te fijas sólo gira a la derecha. Modifica el anterior programa para que gira a la derecha o a la izquierda de forma aleatoria.

# Sigue-lineas

El sigue-lineas son dos sensores que están colocados debajo del robot

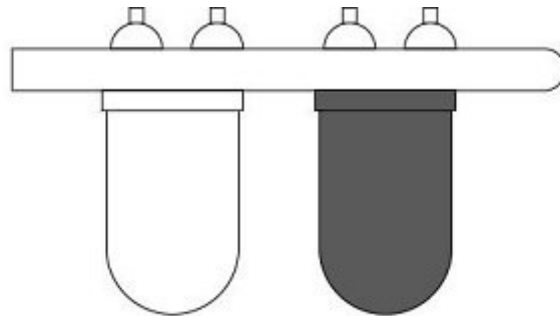


Por unificar criterios, los conectaremos en P3 y P4





Cada sensor tiene dos leds, uno emisor y otro receptor. El receptor recoge la luz reflejada, si hay debajo algo que no refleja la luz (por ejemplo una línea negra) entonces manda OFF en caso contrario ON



**Transmitter**

**Receiver**

**Ground**



OJO VA AL REVÉS es decir \* cuando hay línea negra es OFF \* cuando no hay línea es ON

por lo tanto queremos:

- Que cuando sea OFF sea un 1 lógico (línea)
- Que cuando sea ON sea un 0 lógico (no hay línea)

Esto se llama **CONFIGURACIÓN PULL-UP** (pincha [aquí](#) para saber más) luego lo primero que tenemos que hacer es configurar estos sensores como PULL-UP con estas instrucciones (han traducido **UP** como *subir*):



Están un poco escondidas:



Y luego crear unas variables por ejemplo **izquierda** y **derecha** dentro del bucle que lean esos sensores. El resto del código sólo utilizaremos estas variables:



## Reto 4 Seguir la línea

Otro reto que pide a gritos este sensor:

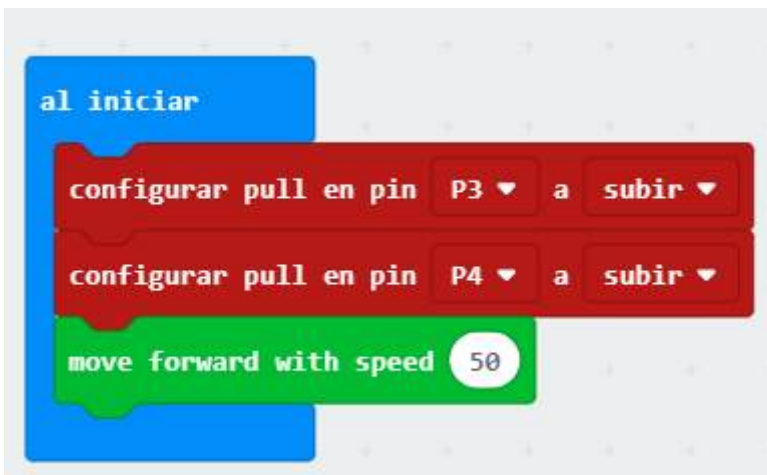
<https://www.youtube.com/embed/cY6nqgHxXio>



Consejo: en clase, antes de atacar con este reto, aconsejamos otro más sencillo como que el robot se mueva y si encuentra línea que se pare, este reto lo puedes ver [aquí](#)

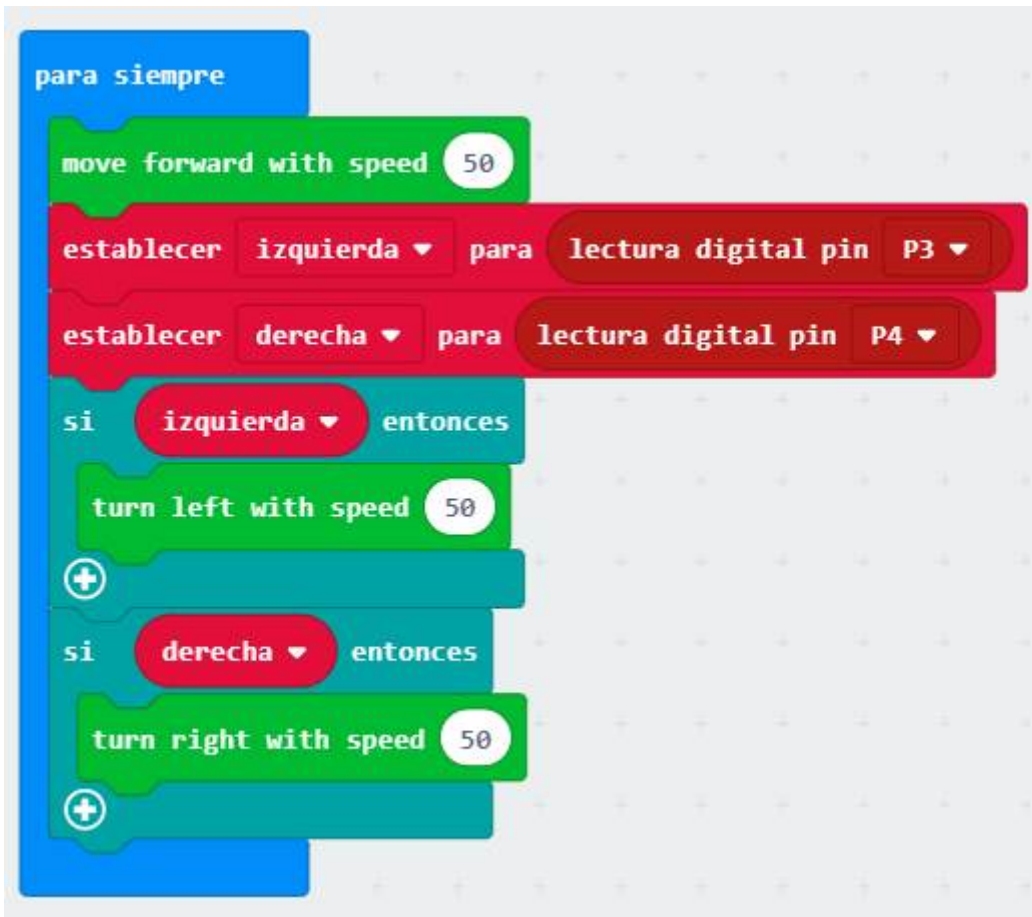
## Descripción del programa

Al empezar configuraremos los sensores en PULL-UP tal y [como hemos explicado](#):



Luego entramos en el bucle donde simplemente dice:

- Sigue hacia delante
- Si te desvías hacia la derecha gira a la izquierda
- Si te desvías hacia la izquierda gira a la derecha



El programa tiene la pega que si aumentamos la velocidad, pierde la línea.

Evidentemente hay muchas versiones y mejoras, por ejemplo [aquí](https://makecode.microbit.org/_U3VP8JhVTXaj) pero ésta [https://makecode.microbit.org/\\_U3VP8JhVTXaj](https://makecode.microbit.org/_U3VP8JhVTXaj) es desde luego la versión más sencilla.



[https://makecode.microbit.org/#pub:\\_U3VP8JhVTXaj](https://makecode.microbit.org/#pub:_U3VP8JhVTXaj)

# Reto 5 Mando a distancia

Esta vez **vamos a utilizar dos micro:BITS**

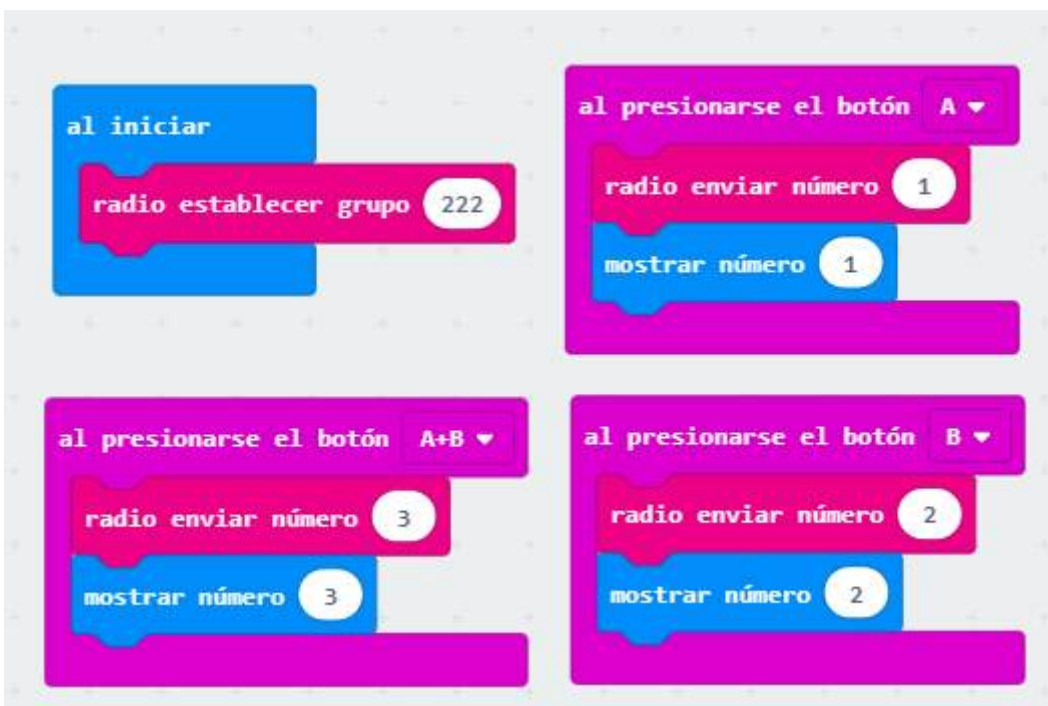
<https://www.youtube.com/embed/oxumk9GYoVU>

## Descripción del programa

### Microbit que hace de mando

Este microBIT hay que alimentarlo con pilas o utilizando una batería típica de móvil.

El mando se inicia en un grupo (en este caso el 222) y simplemente realiza lo siguiente: \* Si se pulsa A manda un 1 y lo visualizo \* Si se pulsa B manda un 2 y lo visualizo \* Si se pulsa A+B manda un 3 y lo visualizo



El programa te lo puedes descargar [aquí](#):

[https://makecode.microbit.org/#pub:\\_cWz6s9aeTPXr](https://makecode.microbit.org/#pub:_cWz6s9aeTPXr)

## Microbit que está en SmartCar

Al iniciar el programa asigna este microbit al mismo grupo de radio que el mando y además asigna una nueva variable con valor 0

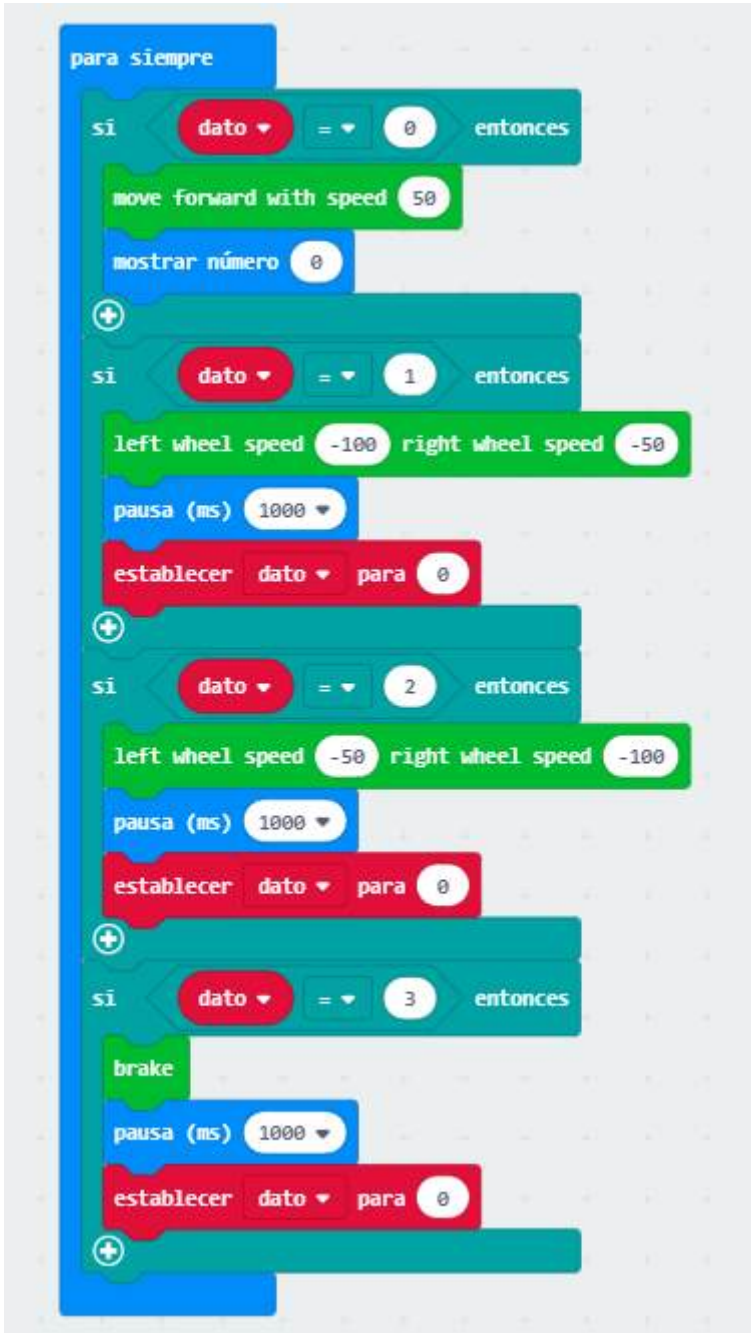


Al recibir un número lo asigna a esa variable y además lo muestra:



Y establecemos un bucle por siempre que :

- Si no ha recibido nada, es por lo tanto dato=0 luego que siga hacia delante
- Si recibe 1 que gire hacia la derecha y hacia atrás
- Si recibe 2 igualmente pero al otro lado
- Si recibe 3 que pare
- En los tres casos anteriores damos un tiempo para que ejecute la instrucción con una pausa y luego reseteamos dato para que siga el robot su camino



```
para siempre
si < dato = 0 > entonces
  move forward with speed 50
  mostrar número 0
+
si < dato = 1 > entonces
  left wheel speed -100 right wheel speed -50
  pausa (ms) 1000
  establecer dato para 0
+
si < dato = 2 > entonces
  left wheel speed -50 right wheel speed -100
  pausa (ms) 1000
  establecer dato para 0
+
si < dato = 3 > entonces
  brake
  pausa (ms) 1000
  establecer dato para 0
+
```

El proyecto te lo puedes descargar [aquí](#) :



[https://makecode.microbit.org/#pub:\\_ftuFv8AReFYq](https://makecode.microbit.org/#pub:_ftuFv8AReFYq)