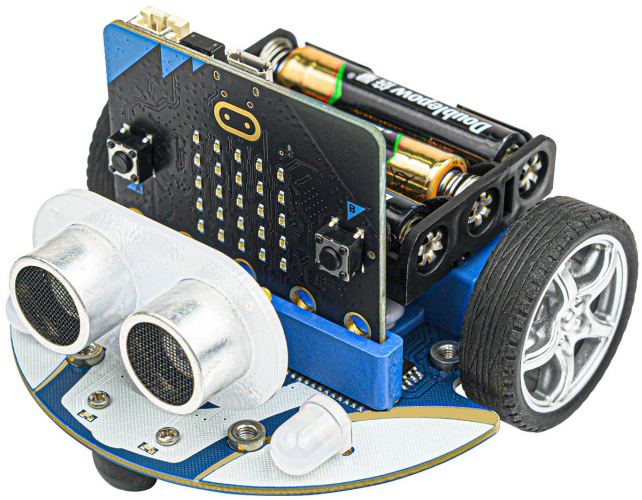
# El robot Cute Bot

# Cutebot se mueve

## ¿Qué es Cutebot?

Cutebot es un **pequeño robot programable** diseñado para enseñar programación y robótica a los niños. Cutebot dispone de una ranura en la que se inserta una tarjeta micro:bit, que actúa como controladora del robot.

El fabricante de Cutebot es Elecfreaks, una empresa con sede en China que desarrolla, fabrica y vende productos educativos basados en micro:bit. La empresa es el distribuidor de micro:bit en China y uno de los **socios de la fundación micro:bit**.



Cutebot tiene las siguientes **funcionalidades**:

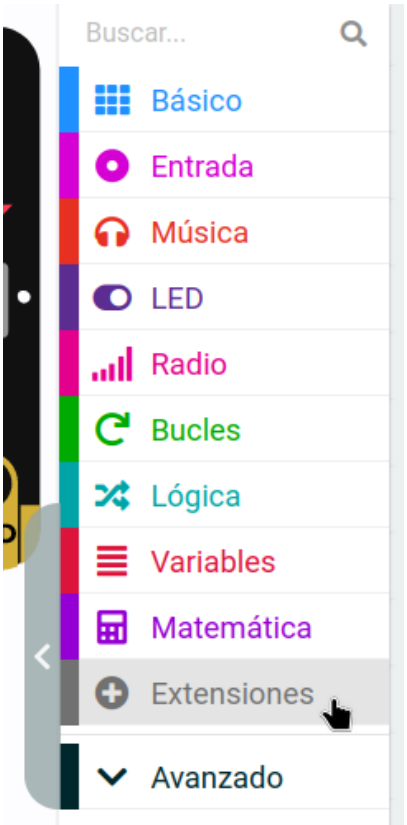
- Dos ruedas controladas independientemente y conectadas a sendos motores provistos de reductoras de engranajes.
- Alimentación mediante tres pilas alcalinas R06 (AAA), tres pilas recargables de NiMH o un pack de batería de litio.
- Dos LED frontales RGB de colores seleccionables desde el código del programa.

- Dos LED RGB bajo la placa para producir efectos de luz mediante una biblioteca de bloques específica.
- Dos sensores colocados bajo la placa para detectar líneas.
- Dos LED azules sobre la placa para informar sobre el estado de la detección de líneas.
- Un emisor y receptor de ultrasonidos para la detección de obstáculos.
- Varios puertos para la conexión de accesorios tales como brazos motorizados, cámaras de inteligencia artificial, pantallas OLED, servomotores, etc.
- Un receptor de infrarrojos que permite el control del robot mediante un mando a distancia.
- Un pequeño zumbador para producir sonidos.

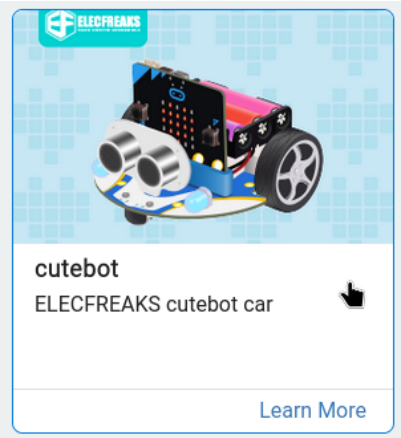
> Cada sensor o actuador es accesible mediante **puertos numerados desde P0 hasta P15**. Por ejemplo, las luces LED situadas bajo la placa se activan desde el puerto P15. Los puertos P13 y P14 corresponden a los sensores del seguidor de líneas.

## Biblioteca de bloques de Cutebot

Para controlar Cutebot es preciso programar previamente la placa micro:bit. Los bloques de programación para el control de Cutebot no están disponibles inmediatamente en el editor MakeCode, por lo que deben ser cargados antes de comenzar a programar. Para ello hay que pulsar sobre el menú **+Extensiones** de MakeCode.

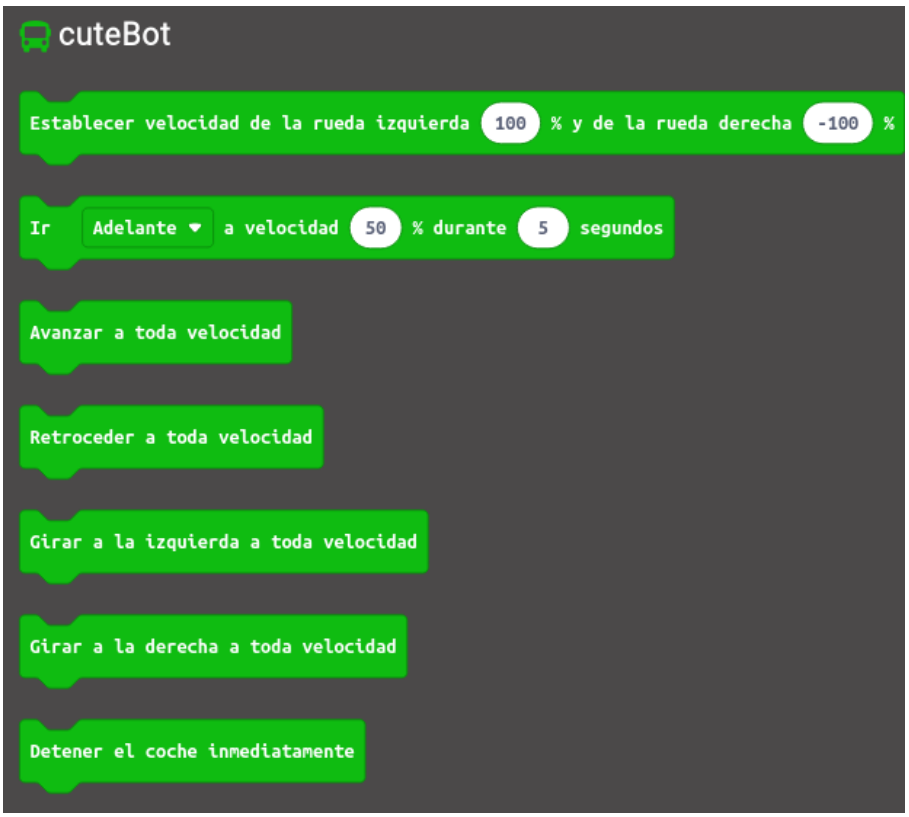


Se desplegará un **mosaico de imágenes de accesorios** entre los que se encuentran varios robots. Tan sólo hay que buscar la imagen correspondiente a Cutebot y pulsar sobre ella.



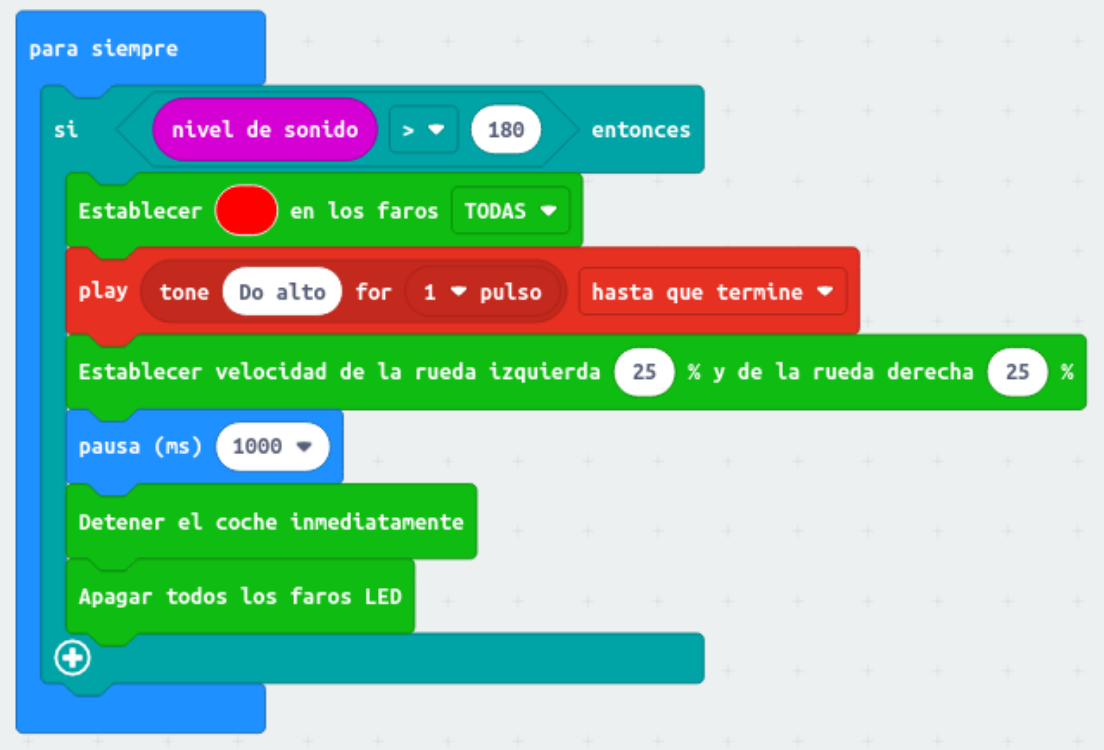

Tras unos instantes, se cargarán **dos nuevos menús de bloques**, uno llamado **Cutebot** para el control del robot y otro llamado **Neopixel** para producir efectos de luces con los LED colocados bajo la placa.

El menú **Cutebot** ofrece **bloques** para el control de la velocidad de las ruedas, para ajustar el color de las luces, para leer los sensores de infrarrojos, de líneas y de obstáculos y para accionar los servomotores que pudieran conectarse.



## Cutebot se mueve

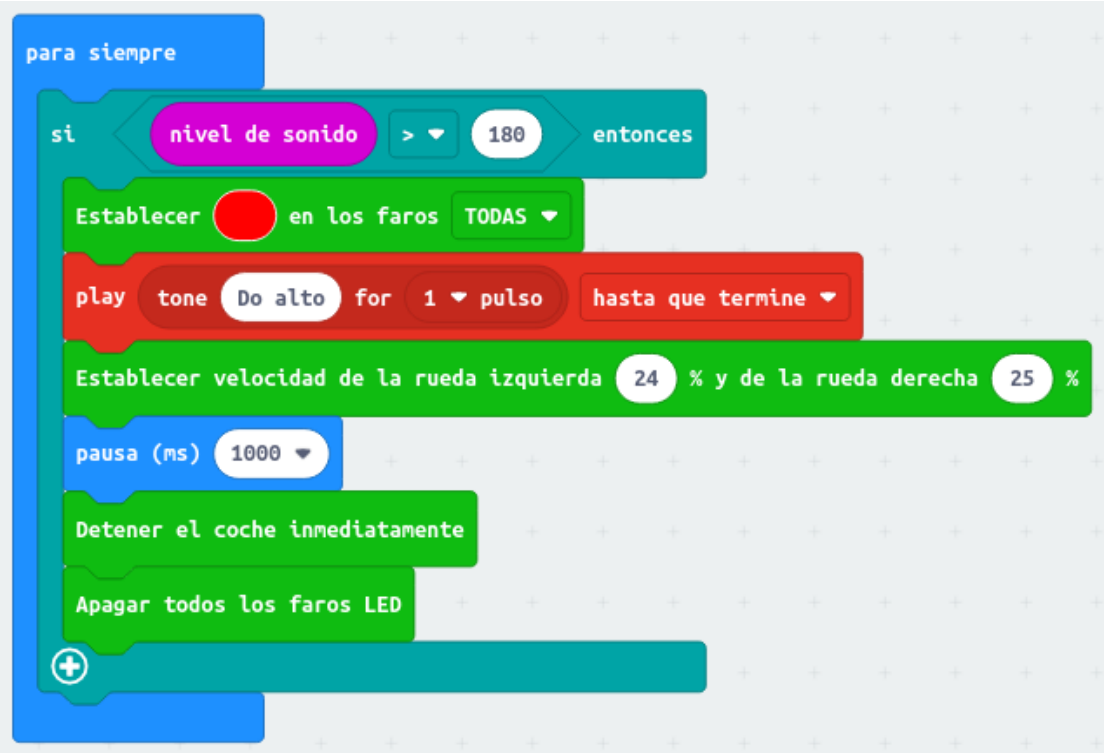
Para probar Cutebot, vamos a programar un pequeño bucle que haga **avanzar una corta distancia al robot cuando detecte un ruido fuerte**, como una palmada. Adicionalmente, el robot encenderá las luces frontales y emitirá un breve pitido de confirmación. El código a introducir es el siguiente:




La secuencia que se ejecuta continuamente dentro del bucle **para siempre** consta de los siguientes pasos:

- Comprobar si el nivel de sonido supera un cierto umbral, en nuestro ejemplo, 180.
- En caso afirmativo encender las dos luces frontales con luz roja.
- Tocar una nota musical breve y esperar a que termine de sonar.
- Hacer girar las dos ruedas hacia adelante a un 25% de su velocidad máxima.
- Esperar un segundo.
- Detener las ruedas.
- Apagar las luces.

En algunos casos, Cutebot no conseguirá avanzar en línea recta. Esto ocurrirá cuando las ruedas giren a velocidades ligeramente distintas, efecto producido por la variabilidad en las características de los motores. Por ejemplo, el robot se desviará a la derecha cuando la rueda izquierda gire a una velocidad algo mayor que la derecha. Para solucionar este problema conviene reducir ligeramente la velocidad de la rueda izquierda o bien aumentar la velocidad de la rueda derecha.



# Lucípeto y Giróvago

## Lucípeto...

Muñoz (2023) presenta un código muy simple para que Cutebot pueda detectar las fuentes de luz intensa y se mueva hacia ellas, al igual que un insecto.

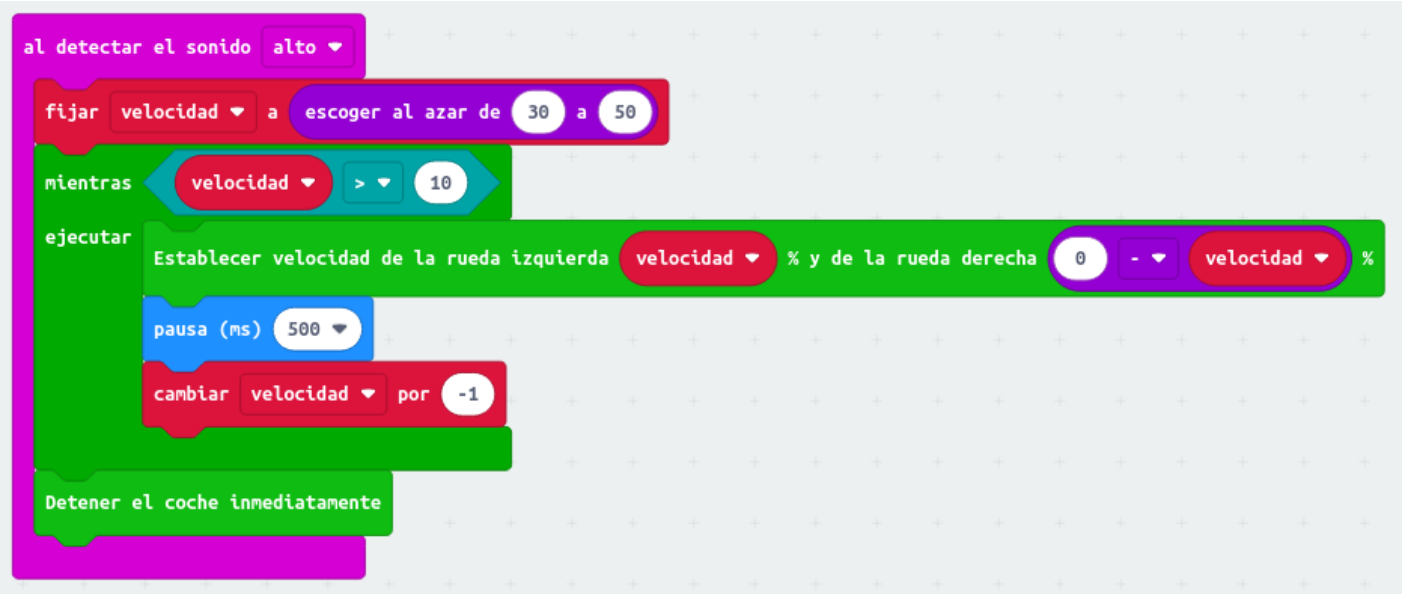
El código combina la lectura del sensor de luz de la placa micro:bit con los bloques de control de velocidad de los motores de CuteBot. Cuando el nivel de luz medido es muy bajo, el robot enciende las luces frontales con color rojo y **gira sobre sí mismo buscando una fuente de luz** lo suficientemente intensa. El giro se consigue moviendo las dos ruedas a la misma velocidad, pero en sentidos contrarios.

Cuando el sensor detecta suficiente luz, las dos ruedas pasan a girar a la misma velocidad, por lo que Cutebot avanza en línea recta hacia la fuente de luz. Adicionalmente, se encienden las luces frontales con luz blanca.


## ...y Giróvago

Vamos a convertir Cutebot en un **spinner activado por un sonido fuerte,** como una palmada. El spinner girará sobre sí mismo, encenderá una luces e irá reduciendo paulatinamente su velocidad hasta pararse.

El código utilizado es el siguiente:




Esta vez introduciremos el código que hace girar al robot dentro del evento **al detectar el sonido alto**, disponible en el menú **Entrada**.

Necesitamos una variable, a la que llamaremos **velocidad**, para guardar el valor de la velocidad de las ruedas. Dicha velocidad irá disminuyendo progresivamente, al igual que ocurre en un spinner; la variable debe ser creada previamente pulsando sobre el menú **Variables**.

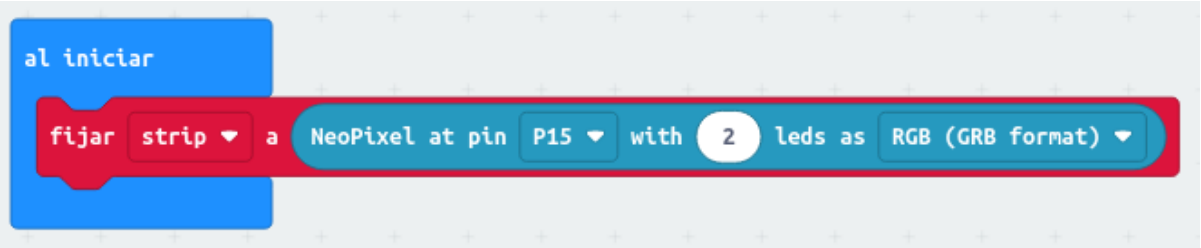

Volviendo al código, cuando micro:bit detecte un sonido fuerte, se activará el evento **al detectar el sonido alto** y la variable **velocidad** tomará al azar un valor inicial entre el 30% el 50%.

Seguidamente se iniciará un bucle **mientras** que mantendrá girando el robot durante 500 mS en cada iteración. La rueda derecha se moverá hacia adelante a **velocidad** y la rueda izquierda se moverá hacia atrás a **-velocidad**.

Transcurridos 500 mS, se restará 1 a la variable **velocidad**. Mientras la velocidad sea mayor que 10, el bucle **mientras** se volverá a ejecutar y Cutebot se mantendrá girando, aunque cada vez a menor velocidad.

Cuando la velocidad sea igual o menor que 10, la ejecución saldrá del bucle **mientras** y parará los motores.

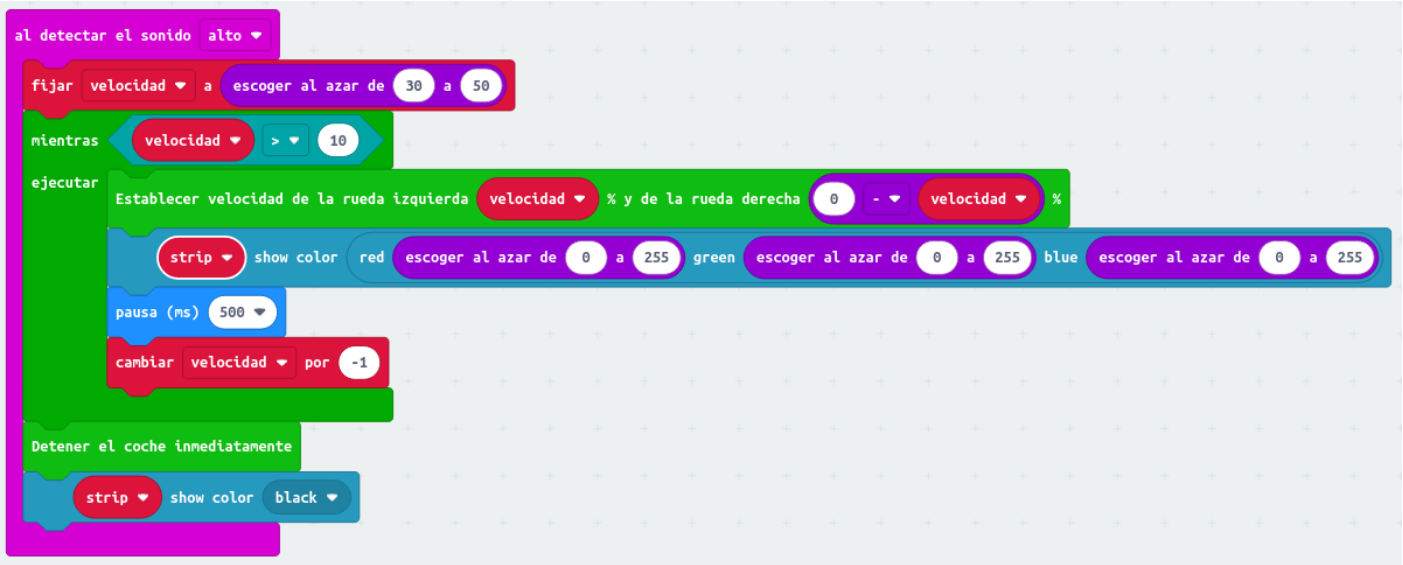
Añadamos ahora al nuestro spinner **efectos de luces** con los LED colocados bajo la placa y con los bloques de la biblioteca **Neopixel**. Esta biblioteca está pensada para producir efectos de luz con tiras de LED. En primer lugar lugar hay que inicializar los dos LED que forman nuestra tira:




**Fijar strip** es un bloque que se encuentra en **Neopixel**. **Strip** es la variable con la que nos refirimos a las luces colocadas bajo la placa de Cutebot, **P15** es el puerto correspondiente a las luces y **2** es el número total de luces a controlar.

Los LED **RGB** generan cada color gracias a la mezcla de tres luces, cada una de ellas correspondiente a uno de los tres colores primarios: rojo, verde y azul.

En cada ejecución del bucle **mientras** se encenderán los dos LED de la variable **strip** con un color que será la mezcla de los tres colores rojo, verde y azul, generados cada uno de ellos con intensidades aleatorias, desde 0 (valor mínimo) hasta 255 (valor máximo).




Al salir del bucle **mientras** es preciso apagar los LED mediante el bloque **strip show color black**.

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

# Evitando obstáculos

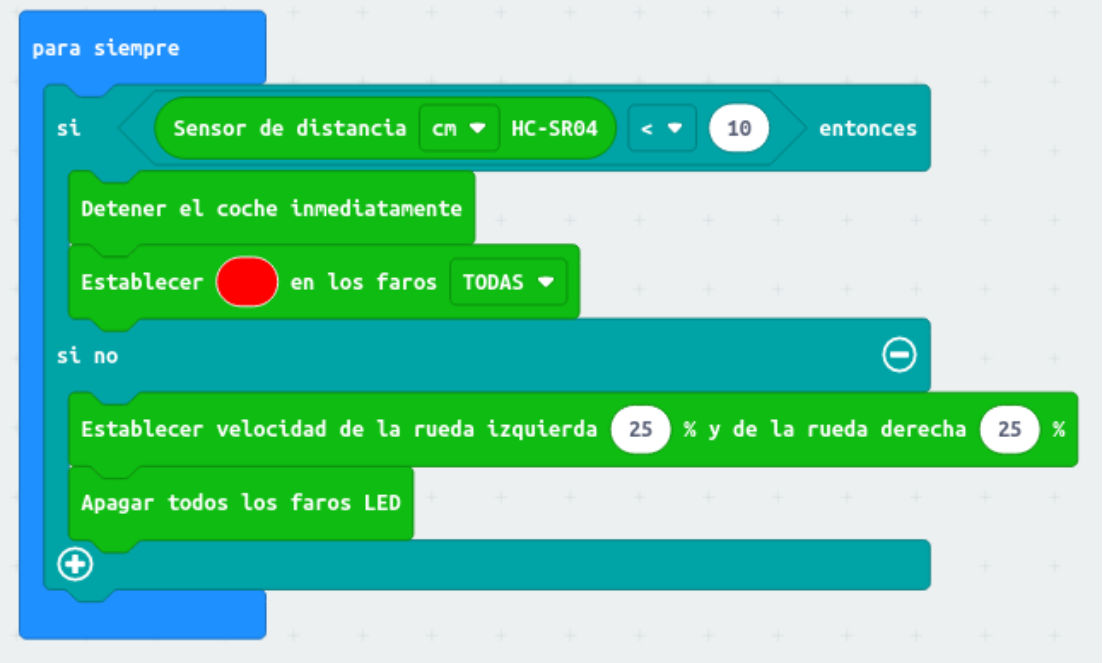
## El sensor de distancia

El kit básico de Cutebot incluye un **sensor de distancia**. Se trata de un accesorio conectable al frontal de Cutebot mediante cuatro pines marcados en la placa como "SONAR". El funcionamiento de la placa se basa en la **emisión de ultrasonidos** y en la recepción de los **ecos** generados cuando las ondas rebotan contra objetos cercanos.

El sensor de distancias debe conectarse en el puerto marcado como "S "IIC". Es muy posible confundirse, ya que ambos conector


El **bloque** que proporciona

la medida de la distancia detectada por el sensor es **Sensor de distancia cm HC-SR04**, y se encuentra en el menú de bloques de **Cutebot**. La medida proporcionada por el sensor puede ajustarse en **cm** y en **pulgadas**.

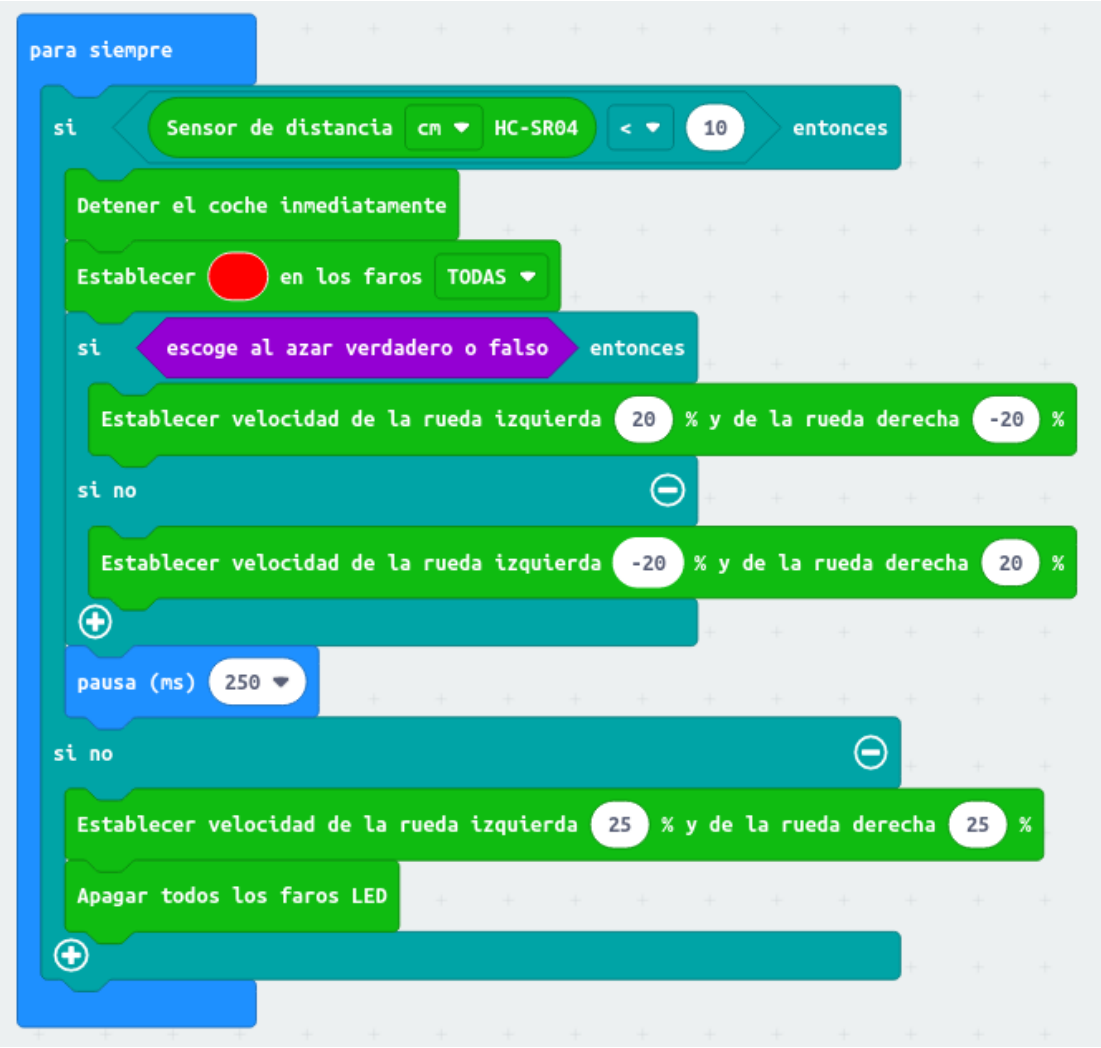



El programa mostrado sobre estas líneas es el código básico que **detiene el movimiento de Cutebot al encontrar un obstáculo**. Cuando el sensor de distancia detecta un obstáculo a menos de 10 cm, el coche se detiene por completo y enciende las luces rojas. En caso de no encontrar ningún obstáculo, el robot simplemente se mueve hacia adelante con las luces apagadas.

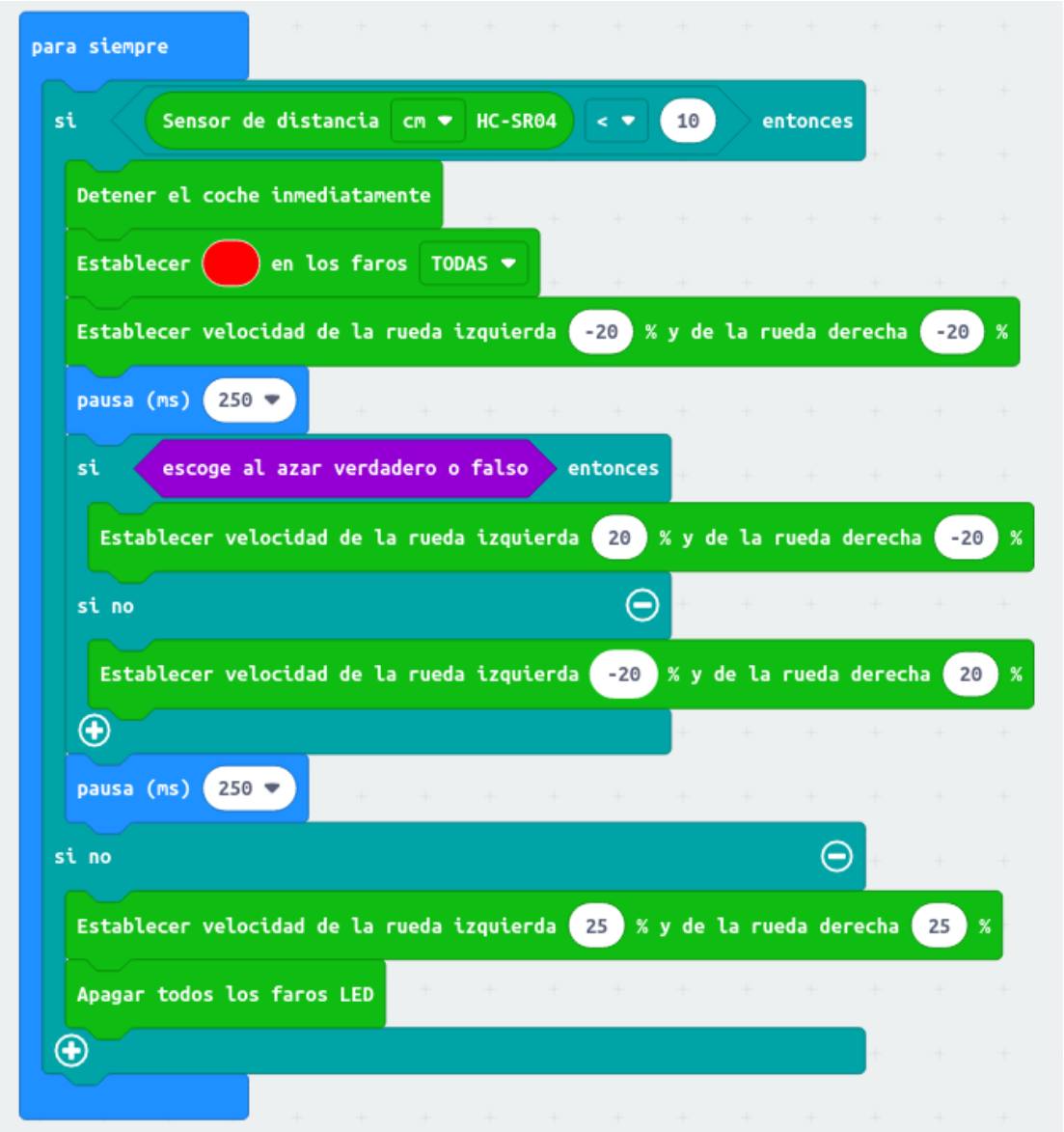
> Recordatorio: si Cutebot no anvanza en línea recta, habrá que aumentar ligeramente la velocidad de una de las ruedas. Si el desvío se produce hacia  la izquierda, habrá que aumentar la velocidad de la rueda izquierda. Si el desvío se produce hacia  la derecha, habrá que aumentar la velocidad de la rueda derecha.

## Detección y evitación de obstáculos

Es posible refinar el código para que Cutebot no se detenga ante un obstáculo, sino que lo esquive. Para ello haremos que cuando el robot se encuentre con un obstáculo, haga un **giro al azar a la izquierda o a la derecha durante 250 ms** y continúe con la ejecución del programa.




Un **ligero retroceso** antes de empezar a girar puede mejorar el comportamiento del robot.




Jugando con la distancia de detección y con los tiempos de giro y retroceso también puede mejorarse el comportamiento del robot ante los obstáculos.

https://www.youtube.com/embed/arC56KgvhIQ

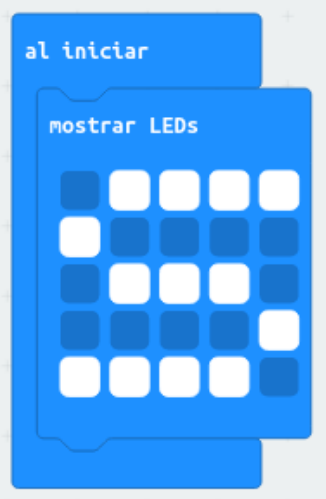
# Un sencillo seguidor de líneas

## Código básico

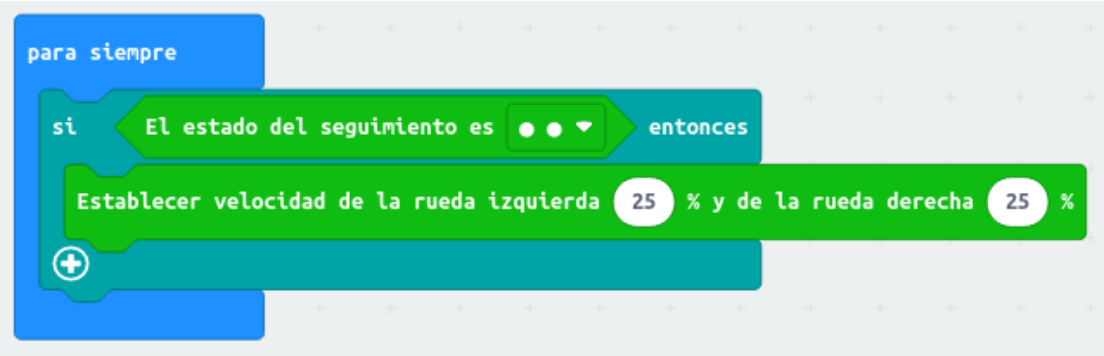
Los dos **sensores ópticos** colocados bajo la placa de Cutebot, similares a los de un ratón de ordenador, pueden detectar líneas negras. El kit de Cutebot incluye una plantilla de papel con una línea negra cerrada a modo de **circuito**. Programaremos micro:bit para que Cutebot sea capaz de **reconocer el circuito y recorrerlo de forma autónoma**.

https://www.youtube.com/embed/515MTWZJaCU

Tras cargar la extensión de Cutebot, comenzaremos por introducir un código que muestre en la matriz de LED de micro:bit el logotipo del **seguidor de líneas**:




El bucle **para siempre** se dedicará a leer continuamente los dos sensores de líneas y a actuar sobre la velocidad de las ruedas. Cuando los dos sensores de líneas detecten el color negro (**El estado del seguimiento es ●●**), significará que Cutebot está justo sobre la línea del circuito y que debe continuar avanzando hacia adelante. Para ello hay que hacer girar las dos ruedas a la misma velocidad.




Si el **sensor derecho detecta color blanco** (**El estado del seguimiento es ●○**), el robot se ha salido de la línea por la derecha  y debe girar hacia la izquierda. Para ello habrá que parar la rueda izquierda y hacer girar la rueda derecha hacia adelante. El código tendrá el siguiente aspecto:

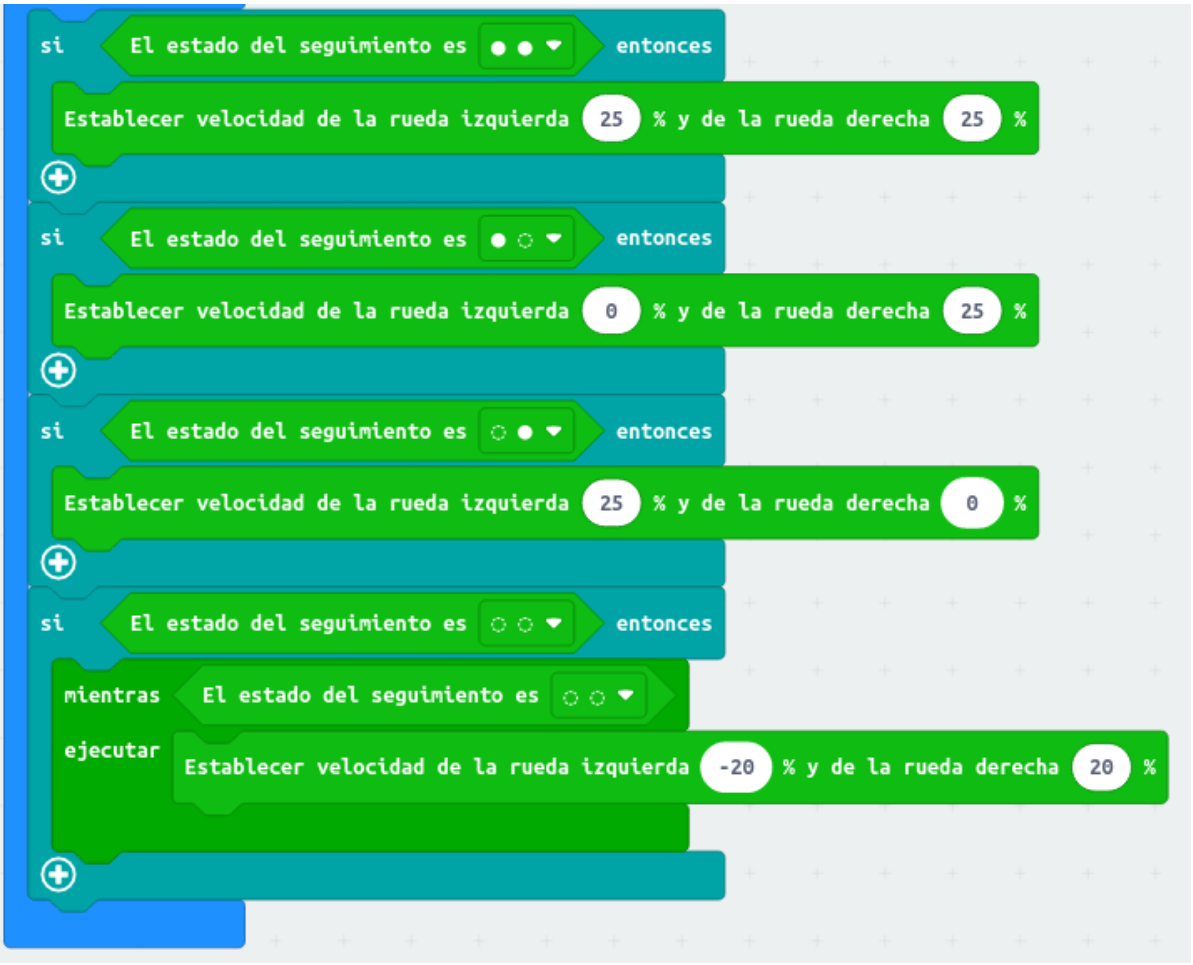

Si, por el contrario, el **sensor izquierdo detecta color blanco** (**El estado del seguimiento es ○●**), el robot se habrá desviado hacia la izquierda y habrá que girarlo hacia la derecha para recolocarlo sobre la línea:


¿Qué ocurre si el robot ha perdido completamente la línea y los dos sensores detectan color blanco? En este caso, (**El estado del seguimiento es ○○**), conviene parar el robot. El código completo se muestra a continuación:


> Cuando las pilas estén muy frescas, Cutebot se desplazará más rápido y puede que pierda la línea continuamente. Este problema se soluciona reduciendo la velocidad de las ruedas, por ejemplo al 20%.

## Mutatis mutandis

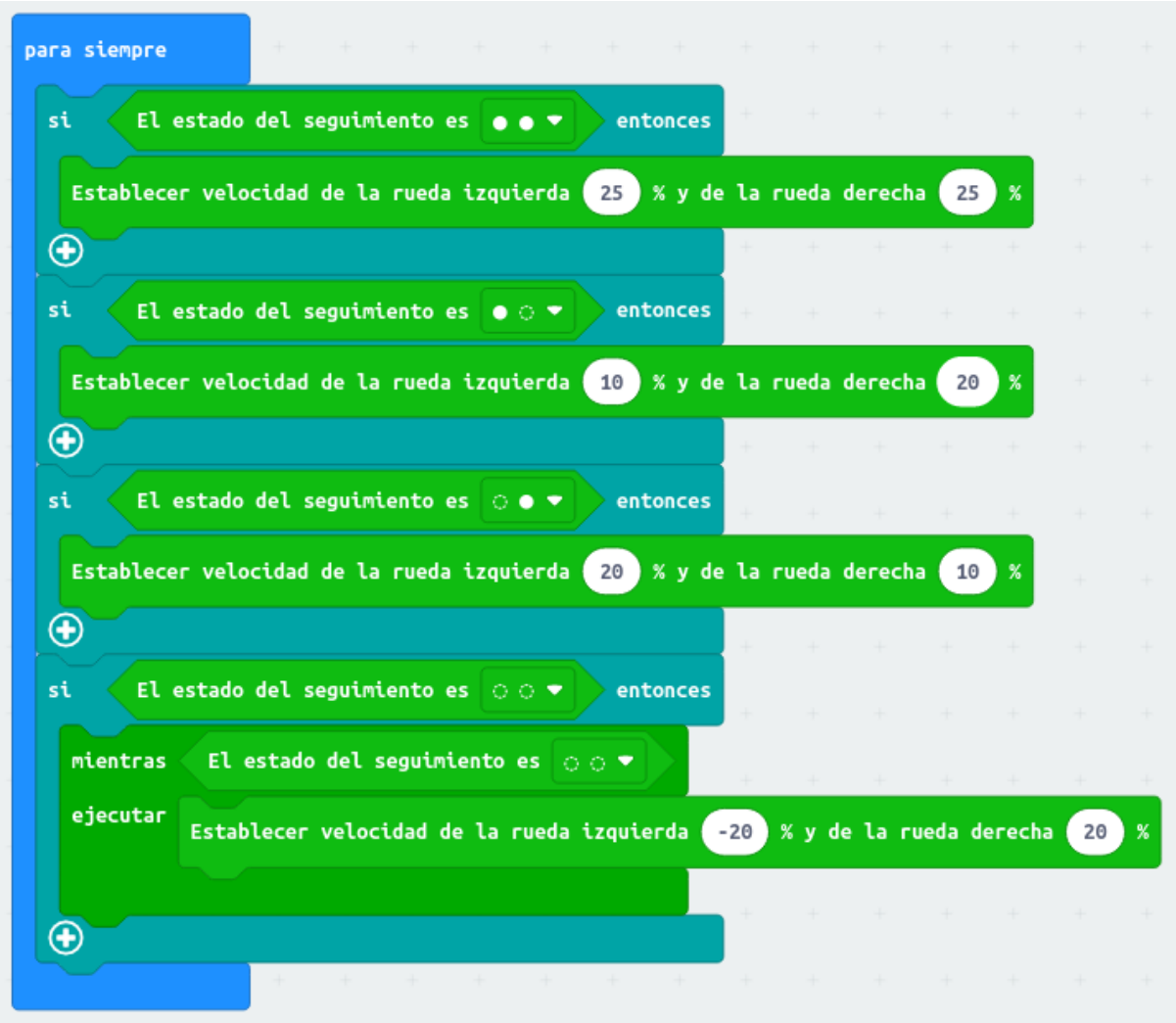
El código desa
intente dar ur
Realicemos u




línea perdida:

De acuerdo con el último bloque condicional, cuando se pierda la línea (**El estado del seguimiento es ○○**) el robot comenzará a rotar sobre sí mismo y no se detendrá hasta que la vuelva a encontrar.

Si no frenamos completamente las ruedas al realizar las correcciones de la trayectoria podremos conseguir que el movimiento de Cutebot sea más suave. El código requiere unos cambios mínimos:




https://www.youtube.com/embed/oX2qu-l3F6Q

## Añadimos funciones automáticas

El seguidor de línea puede incorporar luces automáticas que se enciendan, por ejemplo, al entrar en un túnel. Para ello, añadir al código un **evento de tiempo** que lea el sensor de luz medido, encienda o apague los LED de Cutebot.

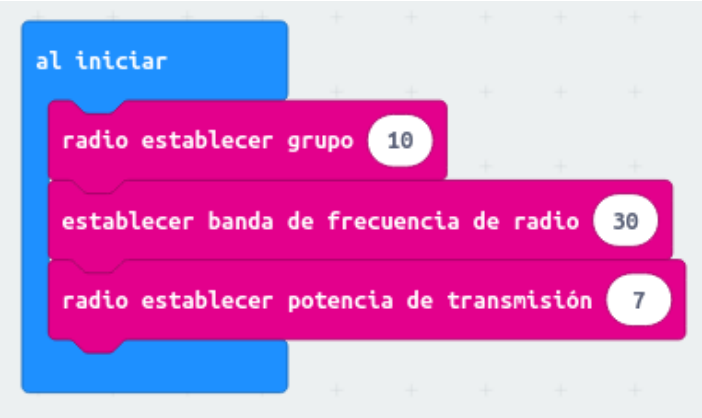
# Rover marciano

## Control remoto

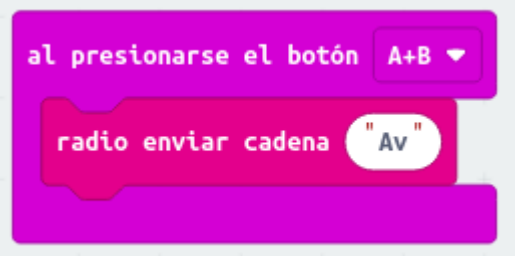
Programemos un **robot de exploración a control remoto**. Esta vez serán necesarias dos placas micro:bit, una receptora montada sobre Cutebot y una emisora a modo de mando de control remoto.

Además, Cutebot enviará a la placa usada como control la información que capten sus sensores durante las exploraciones.

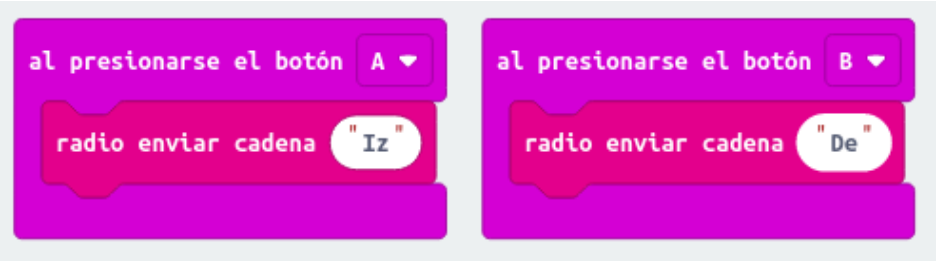
En primer lugar, es preciso que ambas placas ajusten la radio **Bluetooth** en el mismo grupo y banda de frecuencia. Los bloques necesarios se encuentran en el menú **Radio**. El siguiente código es por lo tanto común a la placa emisora y a la receptora.



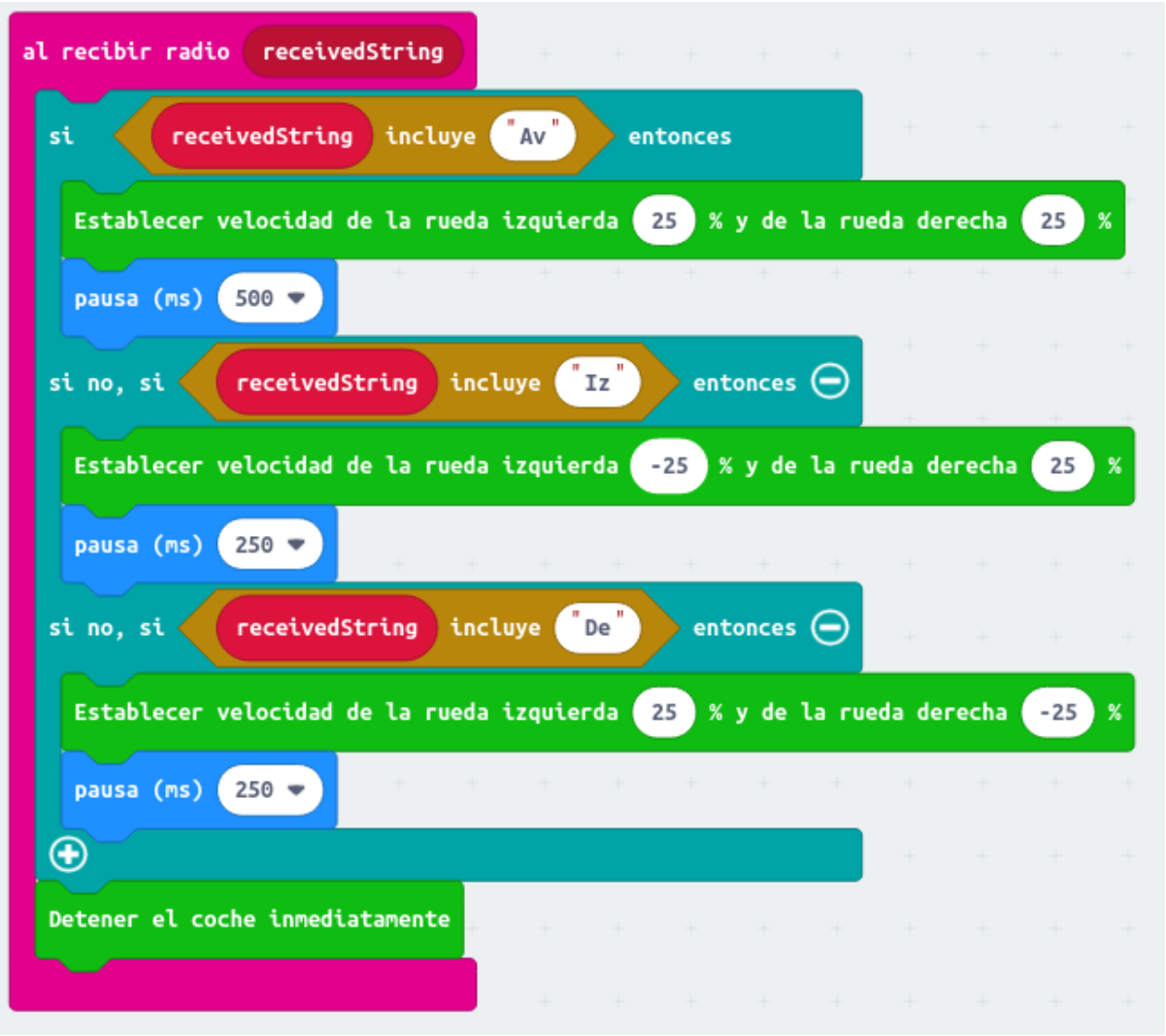

Vamos ahora con el emisor. Cuando **pulsemos los botones A+B**, el rover debe **avanzar**. Para ello le enviaremos por radio una cadena de texto con el código "Av".




Al pulsar el **botón A** el mando enviará un código "Iz", que indicará un **giro a la izquierda**. De la misma forma, Al pulsar el **botón B** el mando mandará un código "De", indicativo de un **giro a la derecha**.




El código básico del emisor está ya resuelto. En cuanto al **receptor**, éste sólo debe ocuparse de mover el rover cuando reciba los códigos. Un rover marciano debe moverse muy despacio, así que cada vez que se reciba un código, las ruedas se mantendrán en movimiento durante unos pocos milisegundos.




La estructura **si...si no, si...** se obtiene del menú **Lógica**; habrá que pulsar sobre los símbolos **+** y **-** para añadir o quitar condiciones a la estructura.

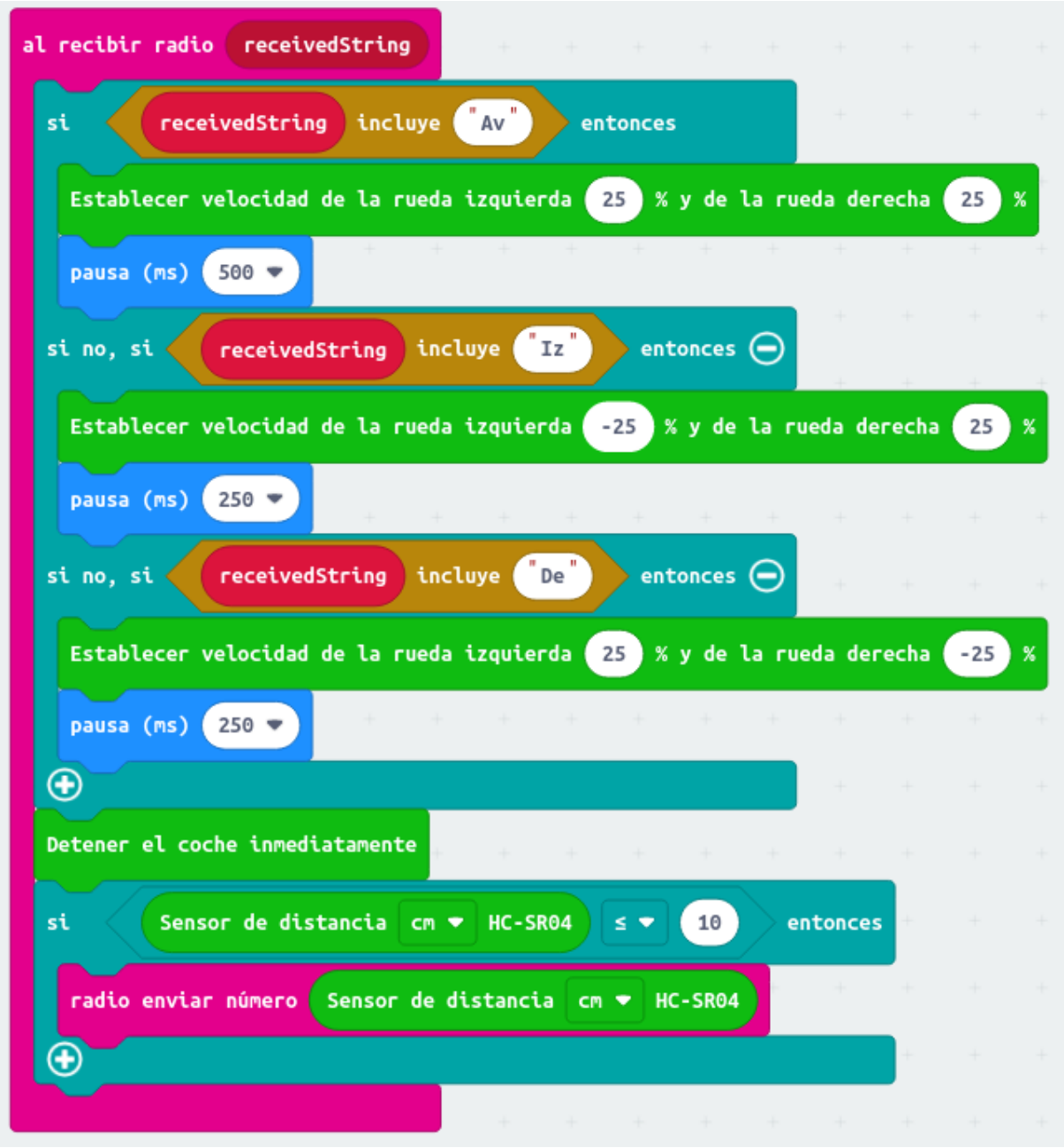
El evento **al recibir radio** se encuentra en el menú **Radio**. **receivedString** hace referencia al código recibido, y puede arrastrarse para encajarse en los comparadores **incluye**, que se encuentran en el menú **^Avanzado** **Texto**.

El código es muy claro: cuando se recibe una cadena de texto se comprueba si contiene el código "Av" y, en caso afirmativo, se hace avanzar a Cutebot durante 500 ms.  Si por el contrario el código es "Iz", el robot girará a la izquierda durante 250 ms. Por último, cuando el código sea "De", el robot girará a la derecha durante otros 250 ms. FInalmente, se pararán los motores.

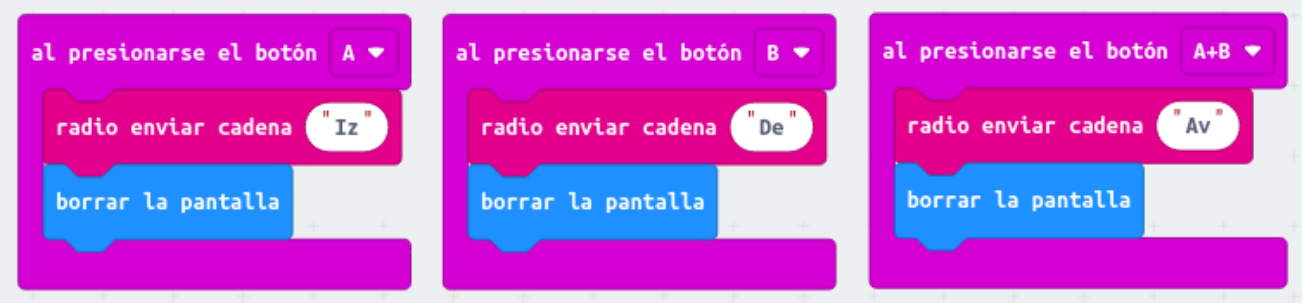
## Transmisión de datos ambientales

Hagamos ahora que el rover transmita información sobre los obstáculos que vaya encontrando. En primer lugar haremos que su placa micro:bit  muestre el logotipo de la misión de exploración planetaria (La Tierra y Marte):

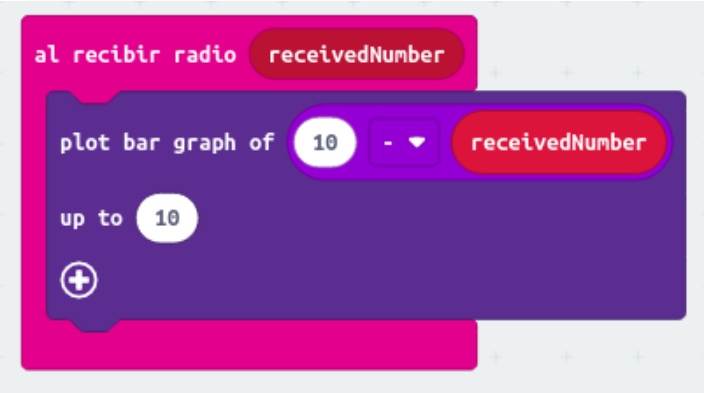

Cada vez que el robot reciba un código para moverse, se comprobará la distancia medida por el sensor y, si ésta resulta ser menor o igual a 10 cm, se enviará el dato numérico al mando a distancia. Como puede verse a continuación, esta función se consigue añadiendo sólo dos bloques más.




Por su parte, la placa que se usa como control remoto debe mostrar en su pantalla el valor de la distancia que haya medido y retransmitido el rover. En primer lugar borraremos  la pantalla LED cada vez que enviemos un código al rover.



Mediante el bloque **plot bar graph** del menú **LED**, mostraremos en la pantalla del mando una barra luminosa cada vez que se reciba un número desde el rover.




Mediante la función de resta entre 10 y el distancia recibida desde el rover conseguimos que la barra muestre su altura máxima cuando la distancia a un obstáculo sea mínima. De igual forma, la barra tendrá un altura nula cuando no haya obstáculos cercanos.

> Podemos conseguir un movimiento más preciso del rover reduciendo los tiempos de avance y giro, por ejemplo, a 250 ms y 125 ms. También podemos añadir la función de marcha atrás al pulsar el logo táctil.