# Registro y radiotransmisión de datos

# La extensión datalogger

**Una de las funciones más interesantes de la placa micro:bit es la de registro de datos** o *data logging*. Gracias a la memoria no volátil de la placa, los datos captados por los sensores pueden ser almacenados cada cierto tiempo para, posteriormente, ser recuperados en formato de hoja de cálculo, accesible como un archivo guardado dentro de micro:bit.

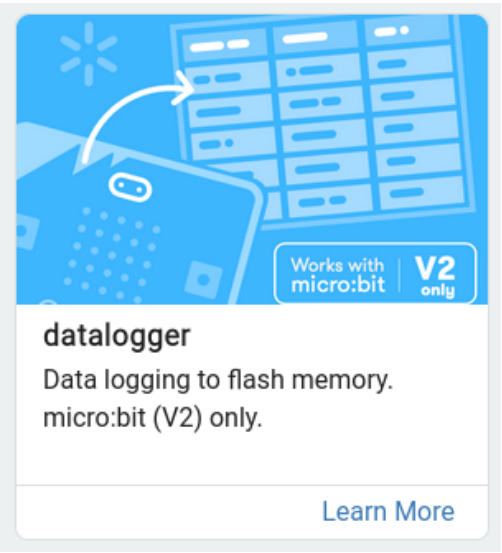
Micro:bit puede convertirse de esta forma en un instrumento de almacenamiento de medidas de mútiples magnitudes físicas, como temperatura, nivel de luz, aceleración, giro, humedad, presión, etc. Teniendo en cuenta que cualquier experiencia científica necesita realizar mediciones, se comprende la **potencia y versatilidad de la placa** en este tipo de actividades.

> El registro de datos sólo está disponible en las versiones 2.x de micro:bit .

Micro:bit puede registrar datos mientras tenga alimentación eléctrica. Con un par de pilas alcalinas R03 podrá funcionar durante algo más de un día. Con una batería externa USB podremos alargar su funcionamiento durante varios días. Usando un cargador de móvil, el registro de datos durará hasta que se llene la memoria de la placa, que es de 512 kB.

La **captura** de datos puede ser **manual**, por ejemplo cada vez que se apriete un botón, o **automática**, programando un evento que registre los datos a intervalos regulares de tiempo. Estos intervalos pueden ser muy largos, como en la medición de magnitudes ambientales como la temperatura, o muy cortos, si es necesario medir fenómenos rápidos como las aceleraciones en una caída.

Para poder usar el registro de datos en MakeCode hay que cargar previamente la **extensión datalogging**. Primero pulsaremos sobre **+Extensiones** y posteriormente sobre el icono de **datalogger**.



Después de unos instantes se habrá cargado la biblioteca, y aparecerá un nuevo menú llamado **Data Logger** en la toolbox de MakeCode.


El nuevo menú incluye bloques para crear, configurar, añadir datos y borrar el **registro de datos o log**. Los datos del **log** se organizan en una matriz de filas y columnas, como en una hoja de cálculo. La primera columna contiene los instantes en los que se realizan los registros. Las siguientes columnas almacenan los valores numéricos registrados.

| time (minutes) | Temperatura | Sonido |
|---|---|---|
| 0.00025 | 21 | 128 |
| 0.2502333333333333 | 21 | 114 |
| 0.5002333333333333 | 20 | 165 |
| 0.7502166666666666 | 19 | 103 |
| 1.00025 | 19 | 183 |
| 1.25025 | 18 | 47 |
| 1.50025 | 18 | 105 |
| 1.75025 | 17 | 143 |

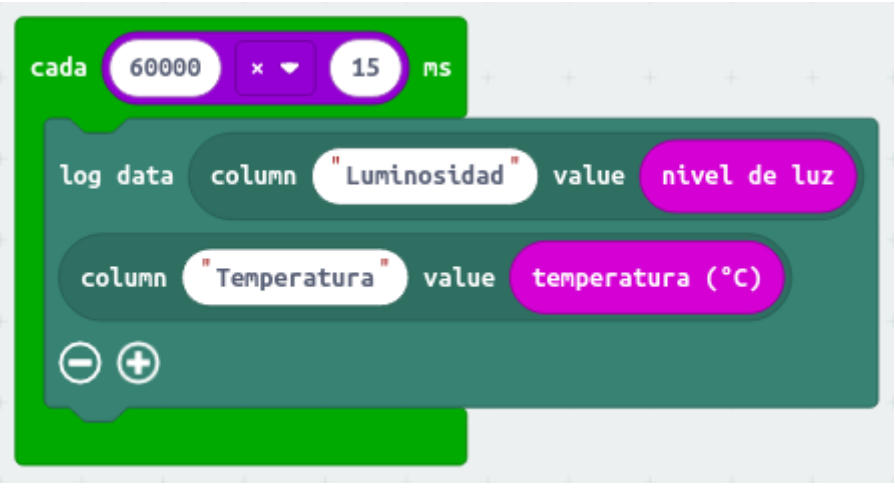
Los datos no se borran al apagar o reiniciar la placa. El **log** puede borrarse desde el programa con el bloque **delete blog** o bien cargando un nuevo programa en micro:bit.

# Registro automático de datos ambientales

Para ilustrar el funcionamiento de la biblioteca **datalogger** vamos a programar un **registrador quinceminutal** de temperatura y luminosidad ambiental.

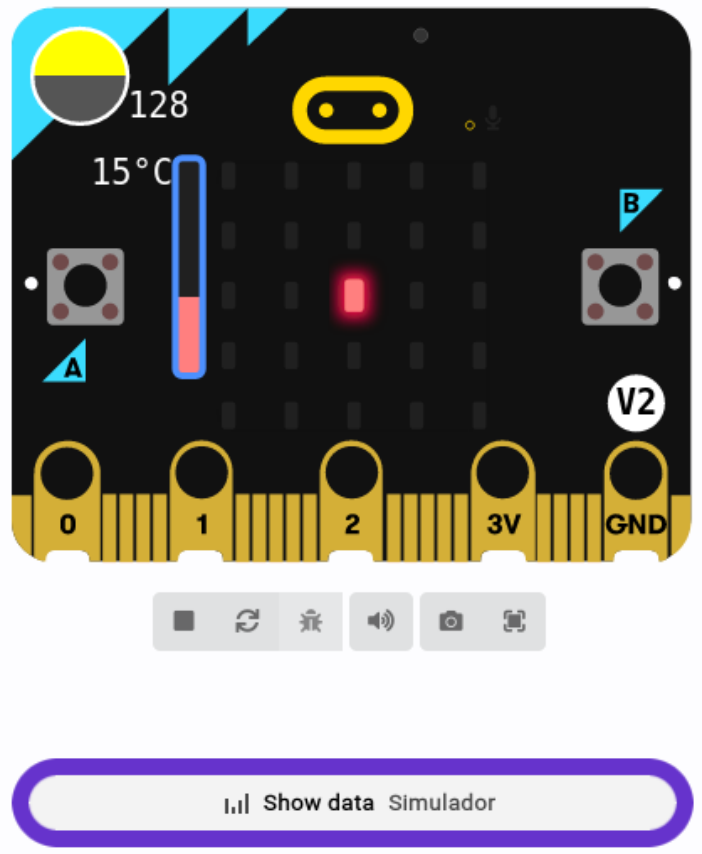
Para inicializar el registro de datos o **log** tendremos que añadir dos bloques dentro del evento **al iniciar**. El bloque **set timestamp hours** añadirá el tiempo, expresado en horas, en las celdas de la primera columna del **log**. **Set columns** creará dos nuevas columnas en el **log**, una llamada **Luminosidad** y otra llamada **Temperatura**. Los botones **+** y **-** sirven, respectivamente, para añadir o quitar columnas de datos.


Cada vez que se ejecute el evento **cada...ms** grabaremos una línea de datos en el **log**. El evento se ejecuta cada 60000 x 15 ms, es decir, cada 15 minutos. El bloque operador **x** se encuentra en el menú **Matemática**. Los valores que van a ser grabados en las celdas de las dos columnas son el **nivel de luz** y la **temperatura (ºC)**, del menú **Entrada**.



Añadiremos un evento para parpadee una luz a modo de testigo o piloto de funcionamiento. Cada segundo, el **LED** central de la matriz alternará su estado entre encendido y apagado. El bloque **invertir x...y...** se encuentra en el menú **LED**.

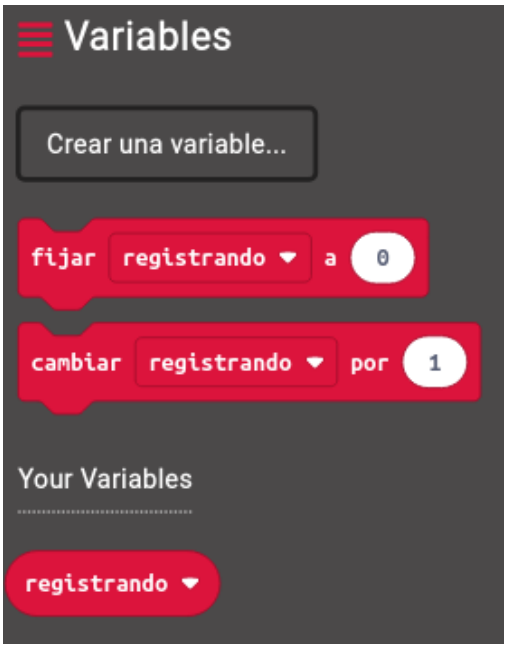

Una vez compuestos los tres bloques de código, el **simulador de Make Code** nos ofrece la posibilidad de asignar valores a las lecturas de la temperatura y del nivel de luz.




Un botón en la parte inferior, etiquetado como **Show data Simulator**, da acceso a los datos simulados del registro.

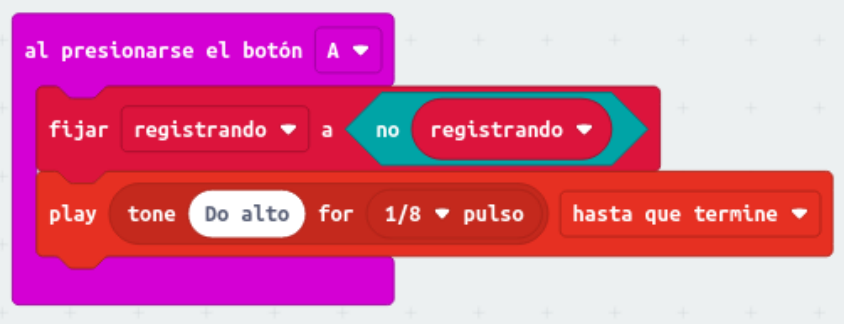
| time (hours) | Luminosidad | Temperatura |
|---|---|---|
| 0.000005 | 128 | 21 |

En ocaciones resulta conveniente **iniciar y finalizar la captura de datos manualmente**, por ejemplo, pulsando un botón. Esta funcionalidad puede conseguirse creando una variable booleana, a la que vamos a llamar **registrando**. esta variable debe crearse desde el menú **Variables**.

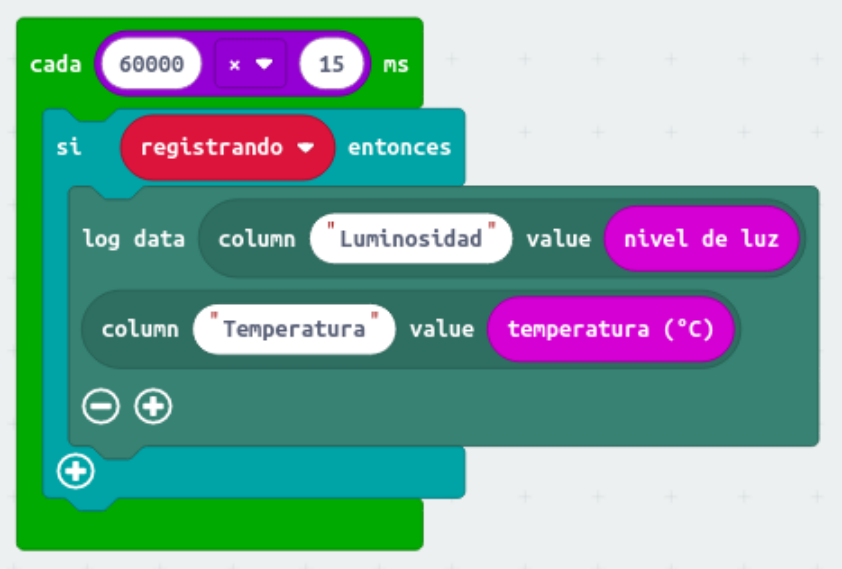



En primer lugar modificaremos el evento **al iniciar** para que asigne el valor **falso** a **registrando**, lo cual indicará que el registro de datos no debe estar activado cuando se encienda la placa. El menú lógica contiene la constante **falso**.


Cada vez que se pulse el botón A, **registrando** cambiará de valor. Si tenía el valor **falso** cambiará a **verdadero** y viceversa. El operador de negación **no**, del menú **Lógica**, lleva a cabo este cambio de valores. Además, haremos que micro:bit reproduzca un **breve sonido** como confirmación del cambio.



Sólo cuando **registrando** sea **verdadero** guardaremos los datos.




Por último, el piloto de funcionamiento del registrador debe apagar el LED central si es que **registrando** es **falso** y mantenerlo intermitente si es **verdadero**.

# Recuperación y tratamiento de datos

Usaremos el programa registrador de datos ambientales, codificado en el apartado anterior, para mostrar cómo se gestionan los datos grabados en la tarjeta.

Los datos mostrados a continuación son los grabados como resultado de colocar la placa micro:bit conectada a su cajita de pilas dentro de un **recipiente hermético de plástico transparente**, y de dejar el recipiente toda una noche en el exterior.

> De acuerdo con la documentación de micro:bit, pueden almacenarse hasta 11000 datos cuando se crea una sola columna. Por otro lado, la autonomía de funcionamiento con dos pilas alcalinas nuevas es de unos dos días. Podemos aumentar la autonomía usando baterías USB o conectando la tarjeta a un cargador de móvil. En la página siguiente usaremos una **biblioteca que nos permitirá ahorrar mucha energía** en los procesos de registro de datos.

Para acceder a los datos hay que conectar la placa al ordenador. El archivo de datos se encuentra haciendo doble clic en la unidad USB MICROBIT, acción que mostrará el archivo MY-DATA.htm. Al hacer doble clic en el archivo, **se abrirá una nueva página en nuestro navegador**:

micro:bit

# micro:bit data log

Download | Copy | Update data... | Clear log... | Visual preview

This is the data on your micro:bit. To analyse it and create your own graphs, transfer it to your computer. You can copy and paste your data, or download it as a CSV file which you can import into a spreadsheet or graphing tool. Learn more about micro:bit data logging.

| Time (hours) | Luminosidad | Temperatura |
|---|---|---|
| 0.00 | 0 | 22 |
| 0.00 | 0 | 22 |
| 0.25 | 0 | 17 |
| 0.50 | 0 | 17 |
| 0.75 | 0 | 17 |
| 1.00 | 0 | 16 |
| 1.25 | 0 | 16 |
| 1.50 | 0 | 16 |
| 1.75 | 0 | 16 |
| 2.00 | 0 | 16 |
| 2.25 | 0 | 15 |
| 2.50 | 0 | 15 |
| 2.75 | 0 | 15 |

Además de la tabla con los datos registrados, la página presenta cinco botones:

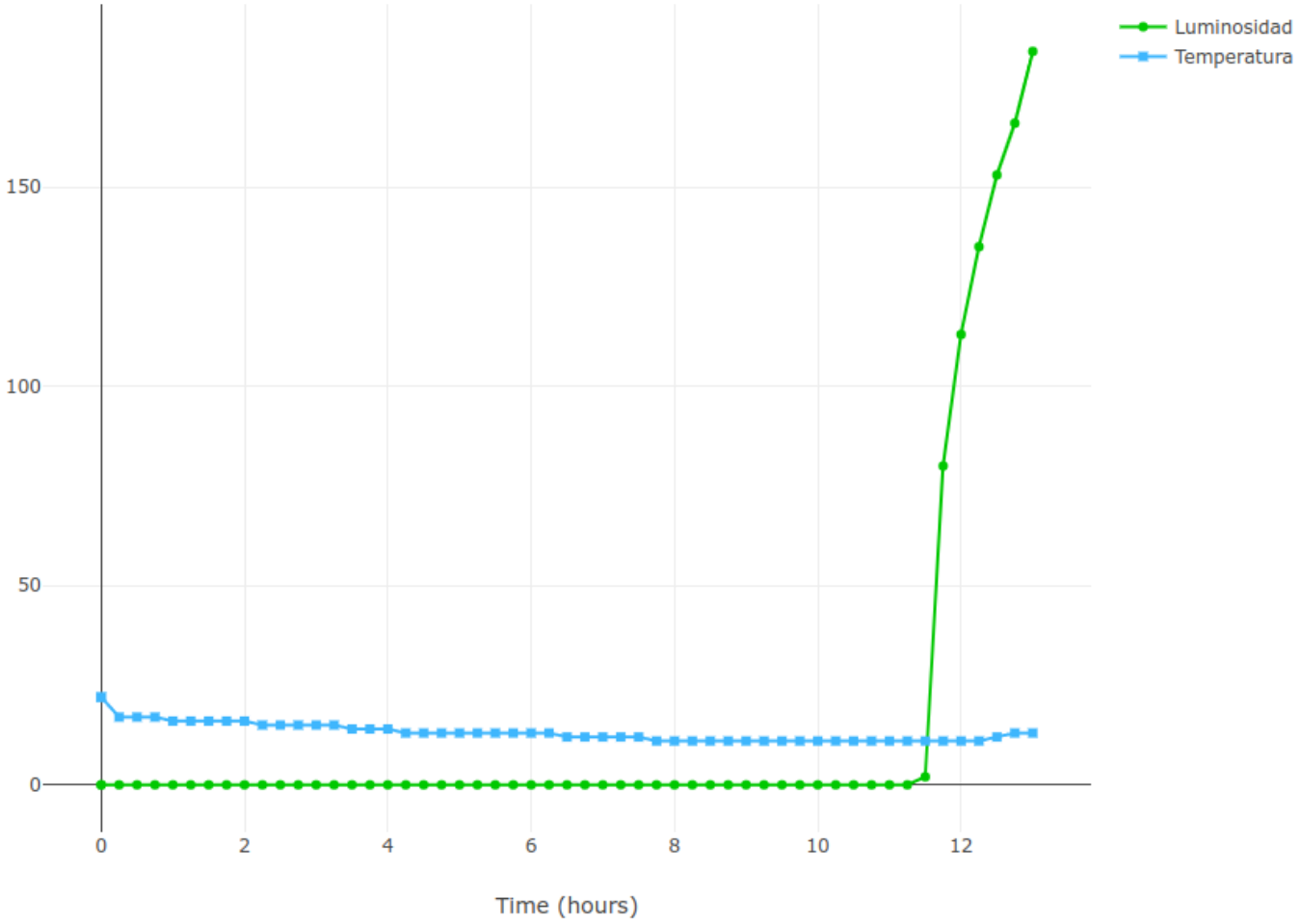
**Download**: descarga los datos en nuestro ordenador en formato *.csv. El documento resultante podrá ser abierto desde cualquier hoja de cálculo.

**Copy**: copia los datos en el portatapapeles. De esta forma podremos pegarlos en una hoja de cálculo o en un procesador de textos.

**Update data...**: actualiza los datos presentados en la pantalla, transfiriendo los últimos datos grabados desde la tarjeta conectada al ordenador.

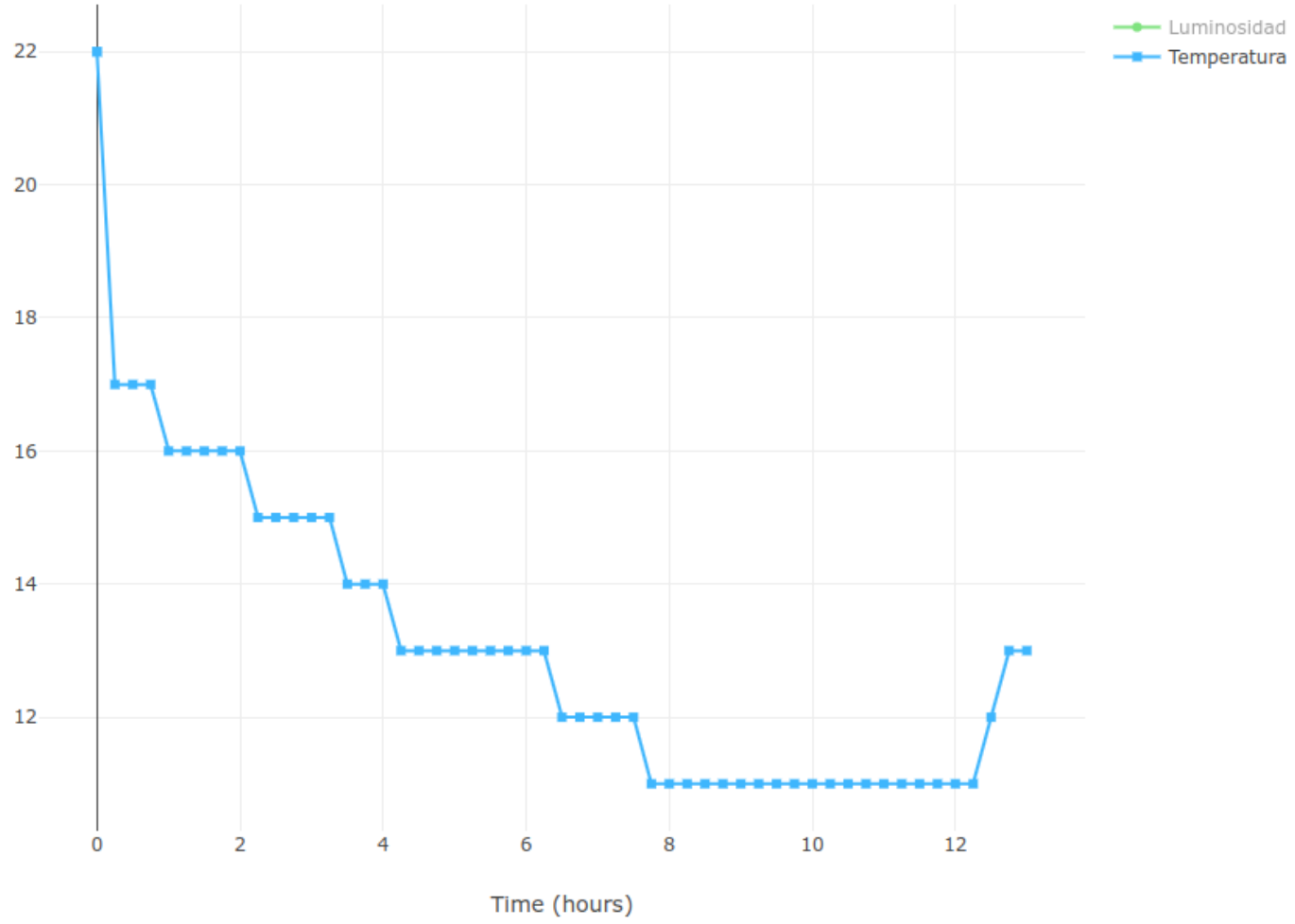
**Clear log...**: borra los datos grabados en la tarjeta.

**Visual preview**: representa gráficamente los datos registrados. En nuestro caso, los datos mostrarán la caída de la temperatura durante la noche y el aumento de la luminosidad a partir del amanecer.




Además, cuando pasemos el cursor sobre la curva, aparecerán **controles suplementarios** para acercar o alejar la gráfica, para reinicializar los ejes y para guardar la imagen. Al colocar el cursor sobre un punto de la gráfica nos serán mostradas sus **coordenadas**:


Podremos también **ocultar o mostrar** la gráfica de una columna determinada haciendo clic en la etiqueta correspondiente, Luminosidad o Temperatura.




El efecto escalonado de la gráfica se debe a que micro:bit guarda valores enteros de la temperatura, sin decimales.

# Extensión Power para el ahorro de energía

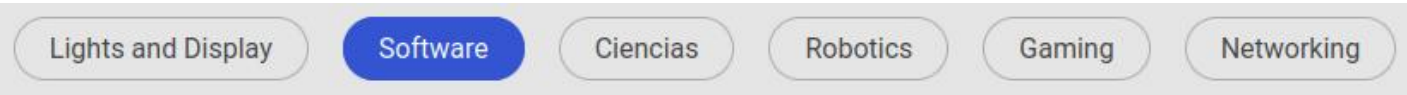
**Power** contiene bloques para poner a micro:bit en estado latente o de hibernación, *sleep mode*, en el cual la ejecución del programa se detiene **reduciendo drásticamente el consumo de energía**.

Micro:bit podrá despertar y volver a ejecutar el programa gracias un evento de tiempo, por la pulsación de un botón, o bien mediante una señal de entrada aplicada a uno de sus pines.

> La extensión Power sólo funciona con micro:bit V2.

**Power** resulta muy útil para crear registradores de datos, ya que éstos pasan la mayor parte del tiempo inactivos y sólo miden y graban magnitudes cada cierto tiempo, a menudo cada muchos minutos.

Para instalar **Power** hay que pulsar sobre el menú **+Extensiones** y, una vez abierta la página de la biblioteca, sobre el botón **Software**.

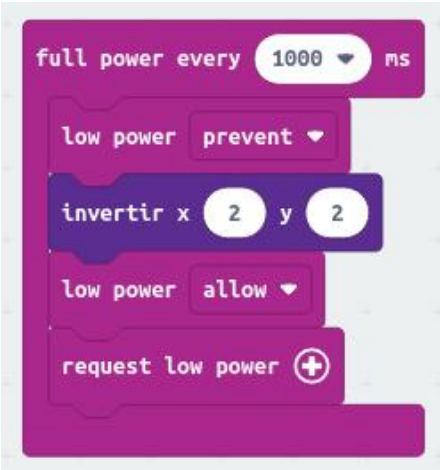



La carga de la extensión requiere pulsar sobre el icono **Power**.


Despueś de la carga de la extensión, **Power** aparecerá el menú de bloques.

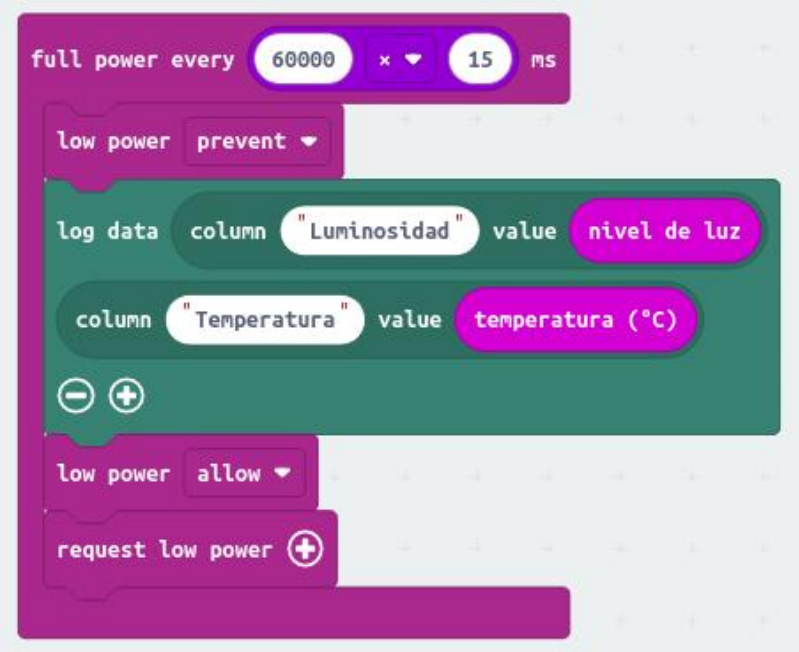
Vayamos con el registrador de datos. Empezaremos modificando el evento **al iniciar** para que micro:bit pase al estado latente justo tras ser encendido. Para ello, tras crear las columnas del registrador de datos, colocamos un bloque **request low power**.


Por supesto, es posible ahorrar energía eliminando el piloto **LED** intermitente, pero vamos a mantenerlo en funcionamiento. El evento de piloto estará ahora controlado por **full power every 1000 ms**. Es decir, despertaremos a micro:bit cada segundo para invertir el estado del piloto **LED**.



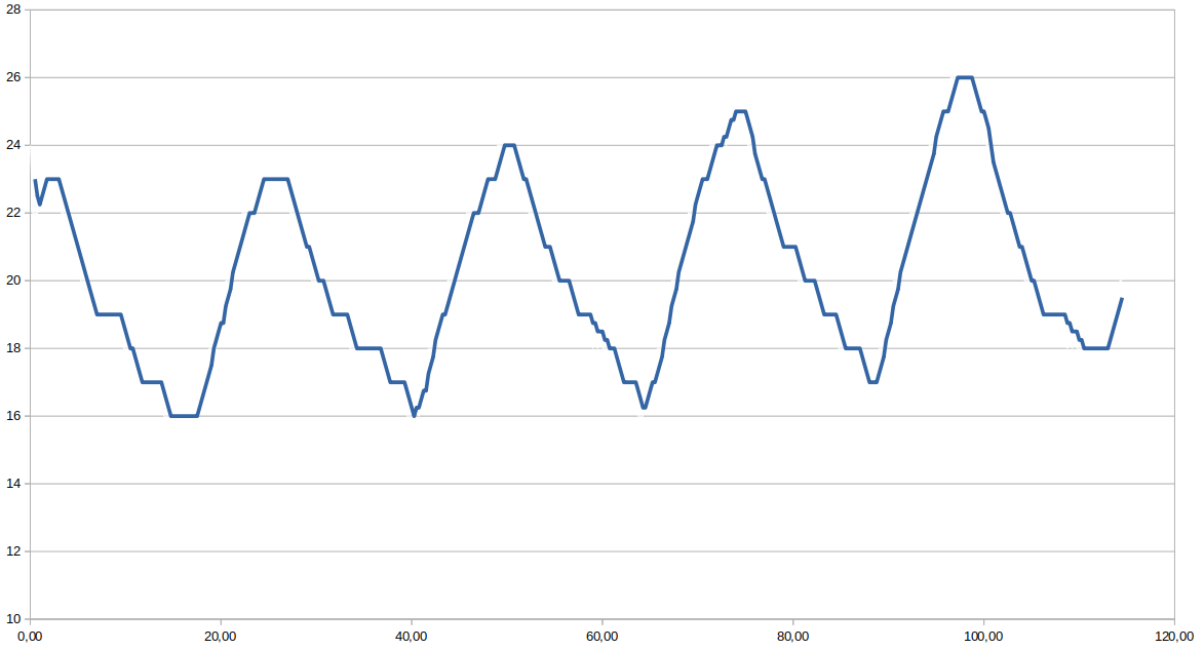

Cuando se despierte micro:bit, habrá que impedir que otro evento que se esté ejecutando lo vuelva a dormir. Esta función la realiza el bloque **low power prevent**, Después de ejecutar este bloque se invierte el **LED** central, se da permiso a la placa para dormir con **low power allow** y se le ordena volver a dormir con **request low power**.

Ahora le toca al evento grabador de datos que se ejecuta cada quince minutos o 60000 x 15  ms. Su estructura es la misma que la del control del piloto. Aquí también hay que prevenir que la placa se duerma mientras grabamos datos.




Podemos **prolongar todavía más la duración de las pilas** manteniendo a la vez la luz piloto para saber si está funcionando la placa. Haremos que el LED parpadee más lento, por ejemplo cada 3 segundos, y durante menos tiempo, por ejemplo durante 75 milisegundos por medio a un bloque **pausa (ms)**. De esta forma, **la placa permanecerá dormida un 97,5% del tiempo**, alargando espectacularmente la autonomía de funcionamiento.


El gráfico siguiente muestra la temperatura exterior en ºC captada por una placa alimentada a pilas durante 5 días de funcionamiento, tras los cuales, la placa seguía alimentada.


28
26
24
22
20
18
16
14
12
10
0,00
20,00
40,00
60,00
80,00
100,00
120,00

# Alarma por radio

Las placas micro:bit disponen de **Bluetooth** integrado de bajo consumo. A máxima potencia y en campo abierto, dos placas pueden comunicarse hasta a 70 metros de distancia. Sin embargo, el alcance dentro de un edificio se reduce a unos pocos metros a causa de las interferencias con otras fuentes de radio, de los muros, de los forjados y del mobiliario.

El menú R**adio** contiene los bloques necesaarios para enviar y recibir datos por Bluetooth. Lo habitual es inicializar la radio dentro del evento **al iniciar** de la siguiente forma:


La **potencia de transmisión** debe tomar un valor entre 0 y 7. A mayor potencia, mayor alcance, pero también mayor consumo de energía.

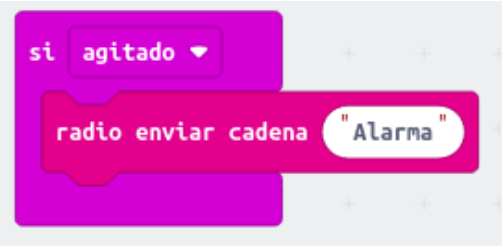
Todas las placas que pretendan comunicarse deben transmitir en el mismo **grupo**. Hay 256 grupos disponibles, numerados desde 0 hasta 255.

También puede ajustarse la **banda de frecuencia**, que puede variar entre 0 y 83.

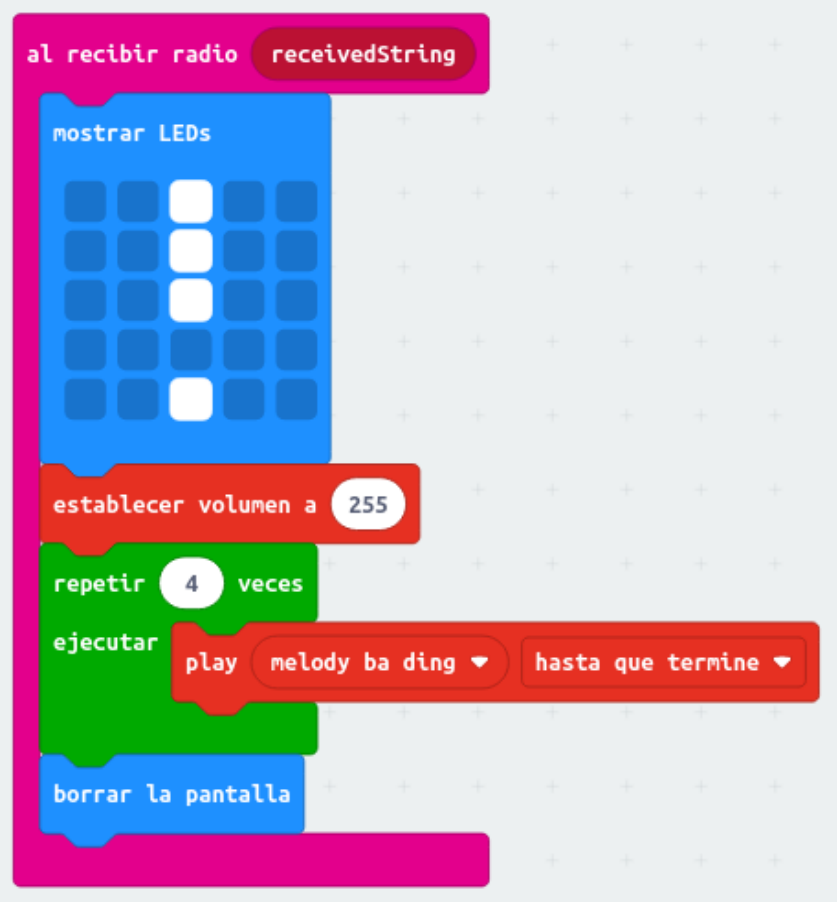
Vamos a comunicar dos placas para crear una alarma inalámbrica por movimiento. Cuando una placa sea movida, enviará un mensaje de texto a la placa receptora y ésta mostrará un signo de alarma en la pantalla a la vez que hace sonar un pitido de alarma.

Las dos placas, la emisora y  lareceptora, deben tener el mismo código en el evento **al iniciar**, con idénticos canal y banda de frecuencia.

La **placa emisora** mandará un mensaje de texto cualquiera, por ejemplo "Alarma", cuando sea sacudida. Usaremos un evento del tipo **si agitado** y un bloque **radio enviar cadena "Alarma"**.




La **placa receptora** usará un evento **al recibir radio receivedString**, que se activará cuando la placa reciba una cadena de texto cualquiera. Dentro del evento mostraremos un signo de admiración, subiremos el volumen a tope y repetiremos cuatro veces un sonido pregrabado. Finalmente borraremos la pantalla.



# Un registrador a distancia de datos ambientales

En esta práctica utilizaremos una placa micro:bit como **sensor de temperatura exterior** y otra placa como **sensor de temperatura interior**. La placa exterior enviará por radio cada 15 minutos una medida de temperatura a la placa interior, que  se ocupará también de medir y registrar la temperatura interior. **Ambas placas trabajaran en modo de ahorro de energía**.

## Programación de la placa exterior

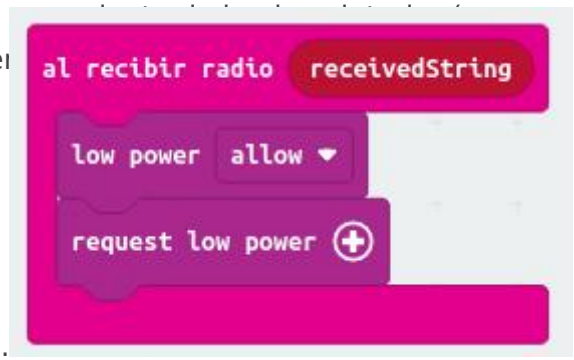
Como primer paso hemos de programar la **placa exterior**, que usará la extensión **Power**, que deberá instalarse desde el menú **Extensiones**. Además debemos configurar la **radio**, por lo que desde el menú de inicio ajustaremos la banda de transmisión, la potencia (máxima) y el grupo, para finalmente poner la placa en hibernación con **request low power**.


Haremos que el **LED central** dé un breve destello de luz de 75 ms cada 5 segundos, para lo cual habrá que despertar a la placa hibernada tras el inicio con **full power every 5000 ms** y volverla a dormir con **request low power**.


**Cada 15 minutos,** o cada 15 x 60 x 1000 = 900000 ms, despertaremos a la placa, tomaremos una medida de la temperatura y la enviaremos por radio. Para despertar a la placa volveremos a usar el bloque **full power every 900000 ms** del menú **Power**.


En este caso nos conviene que el evento anterior encargado de encender el **LED central** cada 5 segundos no nos pueda poner la placa a dormir, por lo que **bloquearemos la hibernación** con **low power prevent**.

Hasta que no recibamos una **confirmación de recepción** cadena de texto cualquiera) no hibernaremos la placa exte




bloqueo de la hibernación y pondremos a dormir a la placa.

> Si va a dejarse la placa exterior a la intemperie, resulta muy conveniente colocarla dentro de un **recipiente hermético**, como un táper.

## Programación de la placa interior

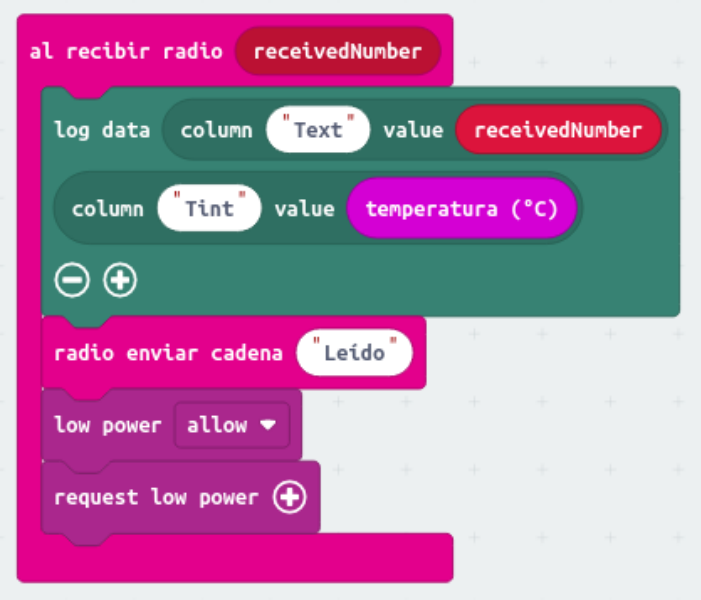
Al igual que en el caso de la placa exterior, habrá que configurar la radio con los mismos parámetros. También tendremos que inicializar el registro de datos, para lo cual habrá que cargar primero la extensión **Data Logger**.


Guardaremos los datos en **dos columnas**, una para la temperatura exterior y otra para la interior. Con **set timestamp** añadiremos una tercera columna que guarde los tiempos, expresados en minutos, en los que realizan las lecturas .

De nuevo, el **LED central** tendrá que parpadear cada 5 segundos a modo de testigo de funcionamiento. El código es el mismo que en el caso de la placa exterior.

Cada 15 minutos despertaremos a la placa y evitaremos que otro evento la ponga en hibernación con **low power prevent**, a la espera de recibir por radio algún dato de la placa exterior.


Cuando la placa esté despierta y reciba por radio un dato de temperatura de la placa exterior, contenido en **received number**, podrá guardar éste junto con la temperatura interior, **temperatura (ºC)**, en el registro de datos.




Por último, mandaremos una cadena de texto cualquiera, en este caso "Leído", a la placa exterior para que pase a hibernación, habilitaremos la hibernación de la plaza interior y la hibernaremos. **Dentro de otros 15 minutos despertarán las dos placas y volverá a comenzar el proceso para registrar un nuevo par de valores**.