Registro de datos

Grabar datos

Para utilizar el Registro de datos con micro:bit V2 tenemos disponibles en Python:

- **import log**. Importamos el módulo, como siempre al principio del código, para tener disponibles las funciones de registro de datos.
- **log.set_labels()**. Para configurar los encabezados de las columnas del registro de datos. Por ejemplo: log.set_labels('temperatura', 'sonido', 'luz').
- **log.add()**. Añadir entradas al registro de datos. Por ejemplo:

```
log.add({
    'temperatura': temperature(),
    'sonido': microphone.sound_level(),
    'luz': display.read_light_level()
})
```

- **run_every()**. Programar entradas de registro en el intervalo especificado de tiempo. Puedes utilizar un programador para registrar datos automáticamente a intervalos regulares. run_every puede utilizarse de dos formas:
  - Como **Decorador** - se coloca encima de la función a programar. Por ejemplo:

```
@run_every(days=1, h=1, min=20, s=30, ms=50)
def mi_funcion():
    # Hacer lo que sea
```

- Como una **función** - pasando la llamada de retorno como argumento posicional. Por ejemplo:

```
def mi_funcion():
    # Hacer lo que sea
```

```
run_every(mi_funcion, s=30)
```

Cada argumento corresponde a una unidad de tiempo diferente y son aditivos. Así, `run_every(min=1, s=30)` programa la llamada de retorno cada minuto y medio.

Cuando se lanza una excepción dentro de la función callback se desprograma la función. Para evitar esto puedes atrapar excepciones con `try/except`.

Los parámetros son:

- `callback` – Function to call at the provided interval.
- `days` – Establece la marca de días para la programación.
- `h` – Establece la marca de horas para la programación.
- `min` – Establece la marca de minutos para la programación.
- `s` – Establece la marca de segundos para la programación.
- `ms` – Establece la marca de milisegundos para la programación.

Ejemplo grabación de registro simple

A continuación vemos un ejemplo de registro:

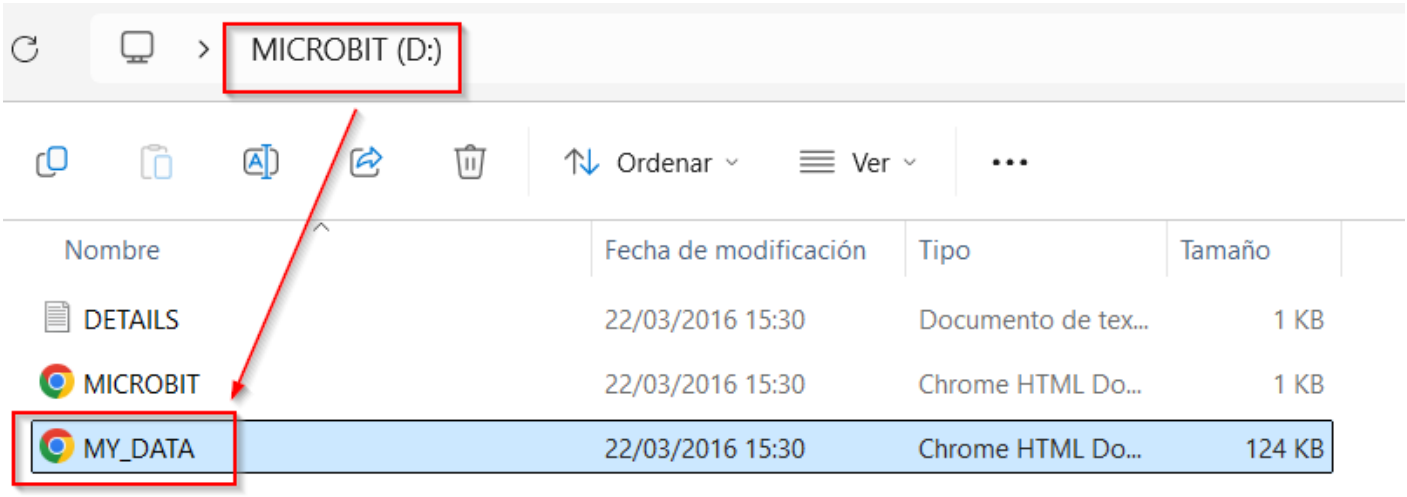
```
from microbit import *
import log

log.set_labels('temperatura', 'sonido', 'luz', timestamp=log.SECONDS)

@run_every(s=5)
def reg_dato():
    log.add(temperatura=temperature(),sonido=microphone.sound_level(),luz=display.read_light_level())

while True:
  sleep(500)
```

https://www.youtube.com/embed/jBympLvqT-8

Lectura de datos

Para ver datos reales grabados en la micro:bit utilizaremos el ejemplo anterior de registro automáticos de aceleraciones.

Una vez que tenemos datos registrados en la micro:bit, la conectamos a un ordenador y dejamos que se monte como una unidad USB de nombre MICROBIT. Si abrimos esta unidad nos vamos a encontrar con tres archivos, uno de ellos es "MY_DATA.HTM".



Si hacemos doble clic sobre el archivo "MY_DATA.HTM" se nos abrirá en una ventana de nuestro navegador por defecto.

micro:bit

micro:bit data log

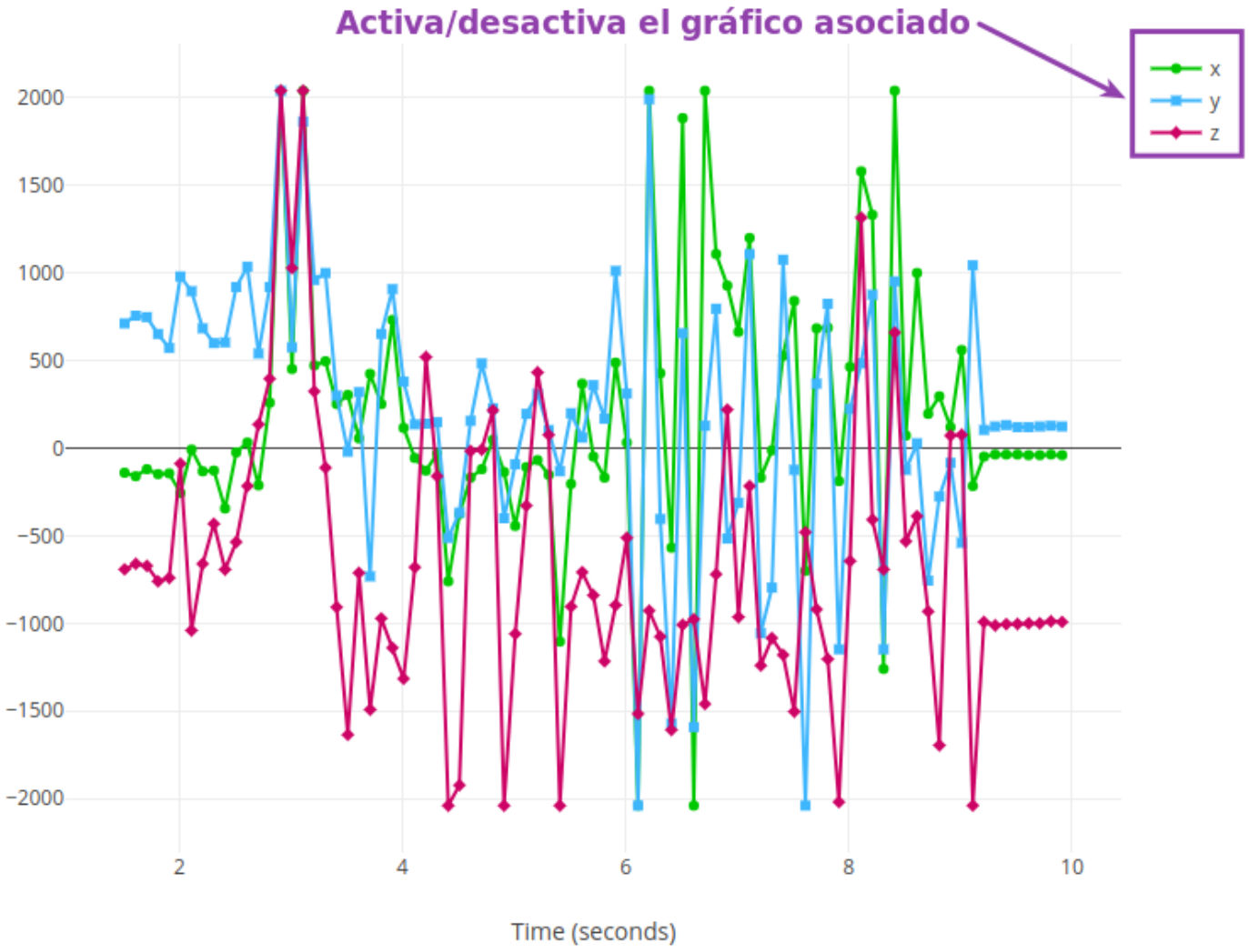
Download | Copy | Update data... | Clear log... | Visual preview

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 (seconds)	temperatura	sonido	luz
6.22	21	0	0
11.19	22	0	0
16.19	22	0	0
21.19	22	72	0
26.19	23	0	0
31.19	23	0	0

Los botones nos muestran diferentes opciones que podemos realizar con estos datos:

- **Descargarlos (Download)**. Se guardan los datos en formato CSV con los valores separados por comas. Estos datos se pueden importar a una hoja de cálculo y realizar todo tipo de análisis con los mismos.
- **Copiarlos (Copy)**. Realiza una copia de los datos en el portapapeles para que podamos pegarlos donde queramos, como por ejemplo en una hoja de cálculo. De esta forma no tenemos que descargar el archivo CSV.
- **Actualizarlos (Update data)**. Comprueba si los datos en la micro:bit han cambiado respecto a la lectura actual desconectando y conectando la micro:bit del puerto USB.
- **Borrar registro (Clear log)**. Nos muestra un mensaje indicando que el registro se borra cuando regrabemos la micro:bit. El programa puede incluir código o bloques para borrar el registro cuando queramos, como es el caso del ejemplo. Por ahora este botón no borra los datos en la micro:bit.
- **Previsualización (Visual preview)**. Muestra los datos obtenidos de forma gráfica. Se pueden mostrar y ocultar utilizando los iconos de la leyenda. En la imagen vemos estos gráficos.



Autor Federico Coca Fuente : Guía de Trabajo de Microbit Licencia CC-BY-SA

Ejemplo más extenso

Vamos a realizar una actividad en la que registraremos la temperatura y la luz ambiente en la misma micro:bit que contiene el programa

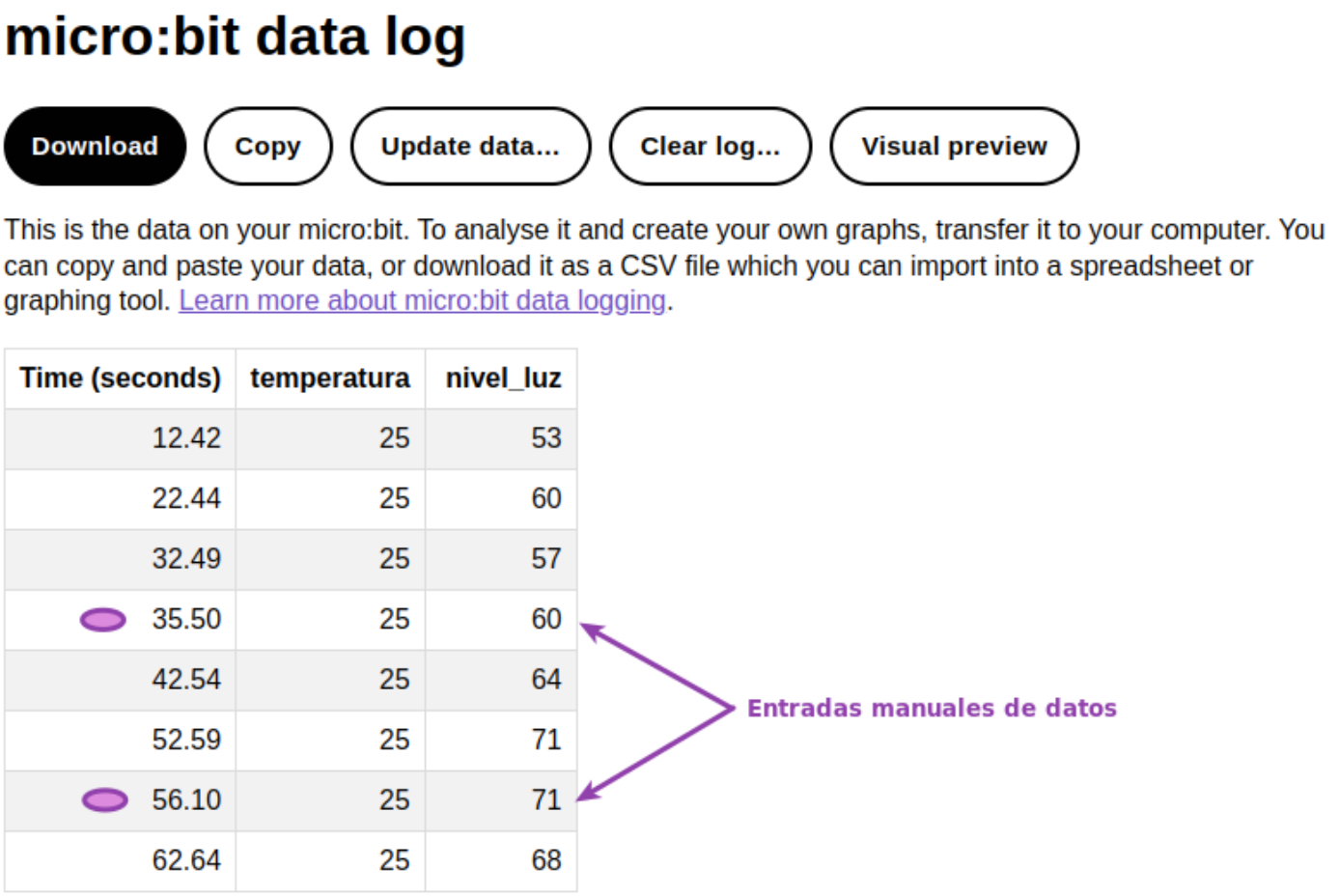
El programa que vemos a continuación realiza un registro automático cada 10 segundos o cuando pulsemos el botón A. Pulsando A+B se borran los datos registrados en la microbit.

```
from microbit import *
import log
```

```
# Configurar etiquetas y establecer la unidad de tiempo
log.set_labels("temperatura", "nivel_luz", timestamp=log.SECONDS)
display.show(Image.NO)
sleep(1000)
# Enviar cada fila de datos a la salida serie
log.set_mirroring(True)

continue_registro = True

# Decorador programado para que se ejecute cada 10s durante 50ms
@run_every(s=10, ms=50)
def reg_dato():
    # Registra cada 10s temperatura y nivel de luz y muestra un icono
    global continue_registro
    if continue_registro:
        display.show(Image.YES)
        try:
            log.add(temperatura=temperature(), nivel_luz=display.read_light_level())
        except OSError:
            continue_registro = False
            display.show(Image.CHESSBOARD)
        sleep(500)

while True:
    if button_a.is_pressed() and button_b.is_pressed():
        display.show(Image.GHOST)
        # Borra el archivo de registro con la opcion "full" lo
        # que asegura el borrado de datos aunque tarde mas tiempo.
        log.delete(full=True)
        continue_registro = True
    elif button_a.is_pressed():
        display.show(Image.YES)
        sleep(500)
        log.add(temperatura=temperature(), nivel_luz=display.read_light_level())
        display.show(Image.HEART)
```

```
    else:
        display.show(Image.NO)
    sleep(500)
```

A continuación vemos el registro de datos tras unos segundos y un par de entradas manuales:



Autor Federico Coca Fuente : Guía de Trabajo de Microbit Licencia CC-BY-SA

> *Página extraída de Federico Coca Guia de Trabajo de Microbit CC-BY-SA*

Revision #7
Created 1 October 2024 19:46:06 by Javier Quintana
Updated 1 October 2024 23:21:53 by Javier Quintana