Datasheet - Programming & Sensors

Program a Micro:bit board to measure sound level

Materials and tools needed

To program a micro:bit board to measure sound level, you will need:

- Micro:bit V2 board and its integrated sensors: this is the main programmable board. It includes a light sensor (via the LED screen), a sound sensor, and an integrated temperature sensor. Approximate price: around €19 per Micro:bit board. (https://www.kubii.com/fr/cartes-micro-controleurs/3091-carte-microbit-bbc-v2-5051259252585.html?mot_tcid=1436612e-e738-4468-b49f-58c52c92a4d4)
- Micro-USB cable: allows you to power the board and program it from a computer.
- External battery (optional): useful for autonomous operation if the board is detached from the computer. The official Micro:bit battery box is available for around €2.20 per unit https://www.kubii.com/fr/alimentations/4237-1913-support-de-pile-officiel-pour-microbit-3272496317253.html?mot_tcid=693572de-fca1-4287-bbd1-df4c014e258b#/appareil-sans.

Various collection methods can be organized:

- ○ **Option 1 (multiple boards):** Use 5 Micro:bit boards to collect data simultaneously from 5 students for one week, then repeat from 5 more students the following week.
- ○ **Option 2 (one board - less expensive):** Use a single Micro:bit board and rotate it among students. Collect data for 2 days from each student, over a total period of 15 days, to obtain a representation of multiple collection points.

You can also buy the Micro:bit V2 kit including the USB cable and the battery box for 21 EUR per kit (https://www.kubii.com/fr/kits-micro-controleurs/3092-kit-microbit-go-v2-5051259252592.html?mot_tcid=e92c2317-81d6-4102-8e90-e56faeb2fe68) or 177 EUR for 10 kits (https://www.kubii.com/fr/kits-micro-controleurs/3093-kit-microbit-club-v2-5051259252615.html?mot_tcid=97a4ea0c-3489-461e-ad35-4aec28defa2d)**Ordinateur ou tablette :** pour écrire et télécharger du code.

- **Programming Environment: MakeCode Online Editor - https://makecode.microbit.org/#editor**

Wiring and Using a Micro:bit Board

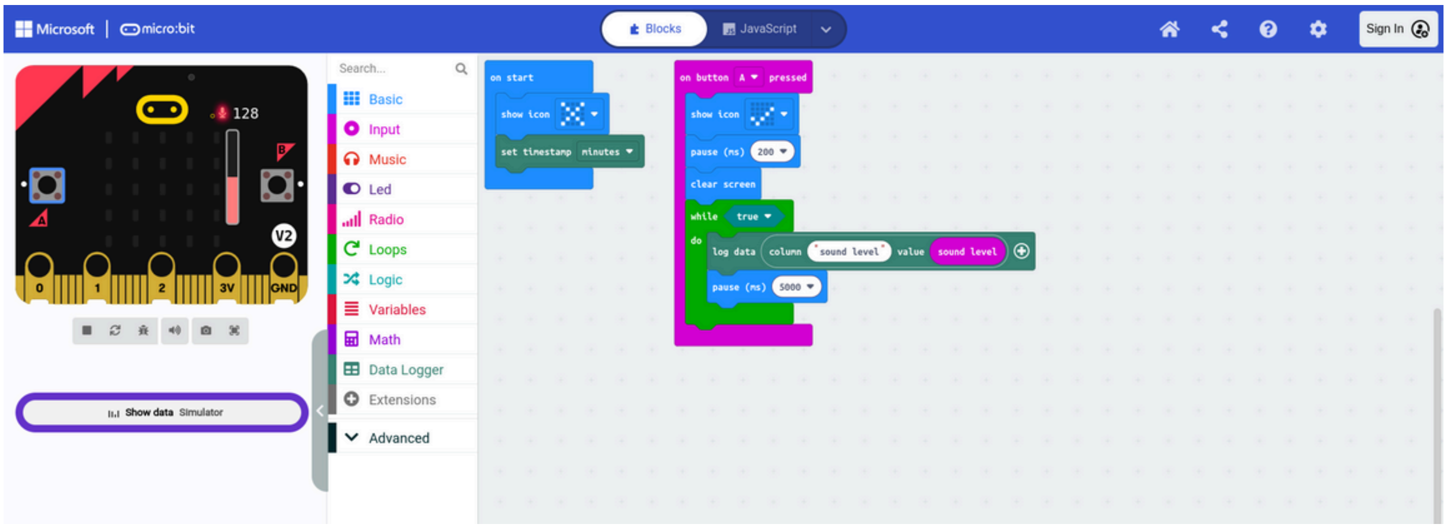
Step 1: Programming the Micro:bit board.

Connect the Micro:bit board: Connect the micro:bit board to the computer on which you created the program using the MakeCode editor. Once connected, the micro:bit board will appear on the computer as a removable disk (e.g., "MICROBIT").

Write the program: Open the MakeCode editor to create a program that collects noise data using the Micro:bit V2 programming board's built-in sensors. Give your project a clear name before you begin.

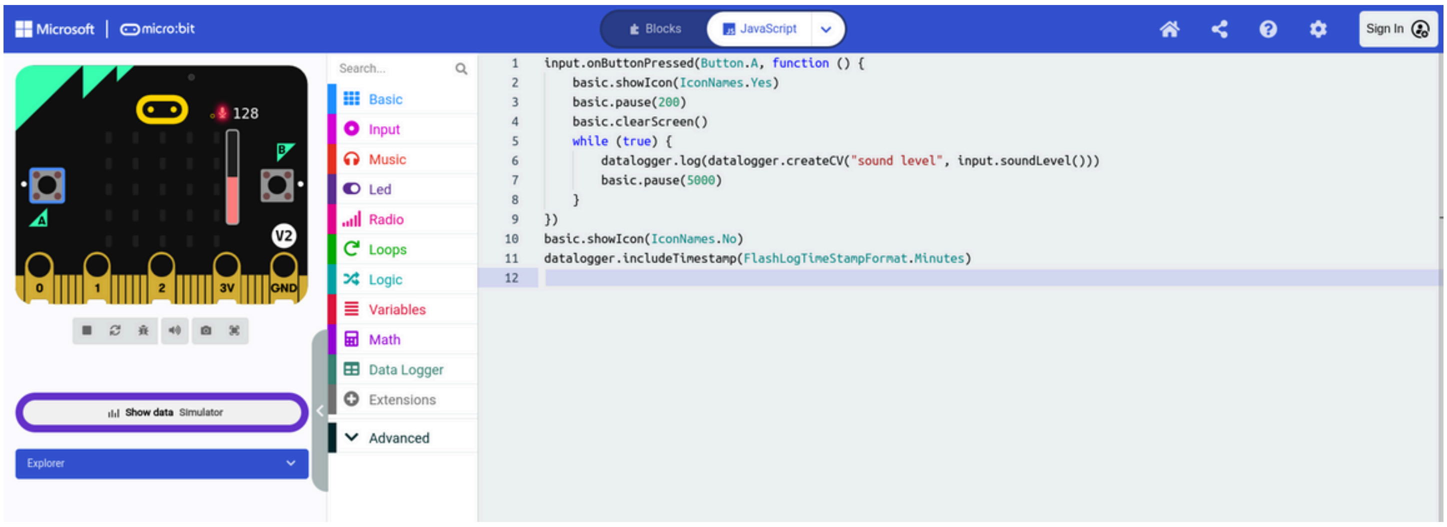
Once in the editor and after creating your new project, you will get the default "out of the box" screen and you will need to install an extension. Extensions in MakeCode are groups of code blocks that are not directly included in MakeCode's base code blocks. Extensions, as the name suggests, add blocks for specific functionality. There are extensions for a wide range of very useful features, adding gamepad, keyboard, mouse, servo, and robotics capabilities and much more. In the block display columns, click the EXTENSIONS button. In the list of available extensions, find the Datalogger extension that will be used for this activity. Click the extension you want to use and a new group of blocks will appear on the main screen.

Then you can start organizing your blocks by following the code provided below (add an infinite loop, save the data to the datalogger...).



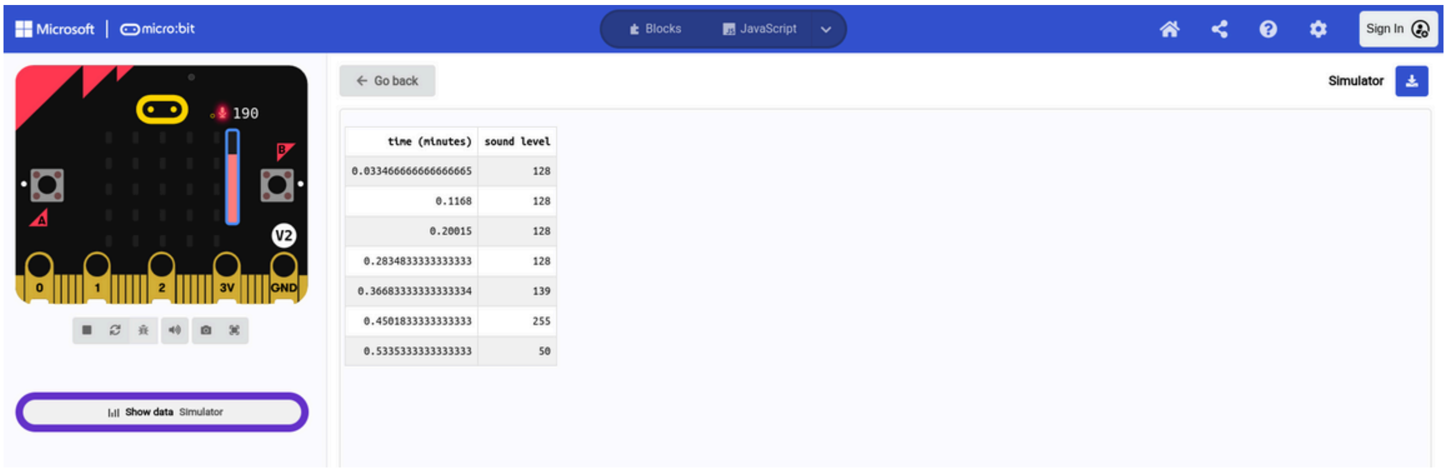
```
on start
    show icon
    set timestamp minutes

on button A pressed
    show icon
    pause (ms) 200
    clear screen
    while true
    do
        log data column "sound level" value sound level
        pause (ms) 5000
```

You can also copy and paste the code into the Javascript editor.



```
input.onButtonPressed(Button.A, function () {
    basic.showIcon(IconNames.Yes)
    basic.pause(200)
    basic.clearScreen()
    while (true) {
        datalogger.log(datalogger.createCV("sound level", input.soundLevel()))
        basic.pause(5000)
    }
})
basic.showIcon(IconNames.No)
datalogger.includeTimestamp(FlashLogTimeStampFormat.Minutes)
```

Test the program using the MakeCode simulator.



time (minutes)	sound level
0.03346666666666665	128
0.1168	128
0.20015	128
0.2834833333333333	128
0.36683333333333334	139
0.4501833333333333	255
0.5335333333333333	50

Once your program is running correctly on the simulator, transfer it to your Micro:bit: Click "Upload" in MakeCode to generate a .hex file. This file contains the compiled program that will enable the board to work. Copy the .hex file from your downloads folder to the "MICROBIT" removable drive. Once the file is copied, the board will automatically reboot and execute the code.

<u>**Step 2: Place the Micro:bit and start recording data.**</u>

Once programmed, place the micro:bit to collect the data you need, for example in a student's garden, in the park next to the school, in the town hall... depending on your choice of location. Use a power bank to ensure your board will collect data throughout the week, for example.

Once positioned, press the "A" button on the MicroBit to start recording data via the program.

<u>Step 3: Retrieving data and preparing the card for the next recording session.</u>

Once the collection period is over, you can retrieve your data from the file called "MY_DATA.HTM", available on the micro:bit reader. Copy it to your computer and rename it with the current date (for example, LOCATIONNAME_YYYY_MM_DD.HTM).

After copying and renaming the file, delete the MY_DATA.HTM file from the MicroBit to free up space and allow for new data recording.

Once opened, the data files will be accessible in HTML format. They will provide all the collected data and allow you to download them in .csv format.

Use and understand the code

Here is the JavaScript code used to program a micro:bit board to regularly collect noise data:

```
input.onButtonPressed(Button.A, function () {
    basic.showIcon(IconNames.Yes)
    basic.pause(200)
    basic.clearScreen()
    while (true) {
       datalogger.log(
       datalogger.createCV("Sound level", input.soundLevel())
       )
       basic.pause(5000)
    }
})
basic.showIcon(IconNames.No)
    datalogger.includeTimestamp(FlashLogTimeStampFormat.Minutes)
```

How the code works.

This program measures the ambient noise level (in decibels) every 5 seconds (the interval can be changed to correspond to 1 minute, 5 minutes, 2 times per hour, etc.) and compiles the information in a "datalogger" from which we can download a .csv file.

A .csv (Comma-Separated Values) file is a text file format used to store tabular data (such as in a table or spreadsheet). Each line in the file represents a row of data, and each value within a row is separated by a delimiter (often a comma, but sometimes a semicolon or a tab). It is possible to retrieve data from a .csv file into a spreadsheet program such as Excel or LibreOffice Calc. In Excel, open the program, click File > Open, select the .csv file, and configure delimiters if necessary using the import tool. In LibreOffice Calc, follow a similar process: click File > Open, select the file, and use the import wizard to choose the delimiter (for example, comma or semicolon). In either case, the data is displayed in table format, ready for analysis.

Initializing the button "A" press event: When the user presses the "A" button on the MicroBit, the function input.onButtonPressed(Button.A, function () {...}) is triggered.

Displaying the "Yes" icon during execution: Before starting data recording, the program displays the "Yes" icon (basic.showIcon(IconNames.Yes)) for 200 milliseconds (0.2 seconds) to indicate that the recording process has started.

Pause for 200 milliseconds: After displaying the "Yes" icon, the program waits for 200 milliseconds using basic.pause(200).

Clear Screen: After the 200 millisecond pause, the screen is cleared with basic.clearScreen(), which prepares the screen for what follows without being cluttered with images.

Infinite data collection loop: The program enters an infinite while (true) loop. This means that data will be collected and recorded endlessly until the MicroBit is turned off or restarted.

Recording data in the datalogger: At each loop iteration, the program records the MicroBit sensor values regarding the sound level using input.soundLevel(), which captures the ambient sound level.

The sensor measures a relative value and does not directly provide values in standard units such as decibels (dB). More precisely, the sensor measures the perceived intensity. This value is a numerical estimate (from 0 to 255), where 0 represents the minimum value (complete silence/total darkness) and 255 the maximum value (very loud noise/intense light).

These values are recorded in the datalogger as variables named "sound level". This is done via the datalogger.log() function:

```
datalogger.log(
datalogger.createCV("sound level", input.soundLevel())
)
```

The createCV function creates a "CV" (context value) for each sensor, and the datalogger.log function saves these values to a file on the MicroBit.

5000 millisecond pause before next reading: After each recording, the program waits 5000 milliseconds (5 seconds) before reading the sensor values again. This is achieved with basic.pause(5000). You can change the pause duration to capture more or less data (e.g., every minute).

Data timestamp (included via datalogger.includeTimestamp): Apart from the button-related function, the datalogger.includeTimestamp(FlashLogTimeStampFormat.Minutes) command is used to include a timestamp with each data record. The timestamp format is in minutes, meaning each record will have a time indicator based on the minutes elapsed since the program was started.

Displaying "No" icon before execution: Before the user presses the "A" button, the program displays a "No" icon (basic.showIcon(IconNames.No)) to indicate that the MicroBit is waiting for user action.