



# Programmer une carte Micro:bit pour mesurer le niveau sonore

## Matériel et outils nécessaires

Pour programmer une carte micro:bit pour mesurer le niveau sonore, vous aurez besoin de :

- **Cartes Micro:bit V2 et ses capteurs intégrés** : La carte programmable principale incluant un capteur de niveau sonore intégré - Environ 19 EUR par micro:bit ([consultez les prix ici](https://www.kubii.com/fr/cartes-micro-controleurs/3091-carte-microbit-bbc-v2-5051259252585.html?mot_tcid=1436612e-e738-4468-b49f-58c52c92a4d4) - [https://www.kubii.com/fr/cartes-micro-controleurs/3091-carte-microbit-bbc-v2-5051259252585.html?mot\\_tcid=1436612e-e738-4468-b49f-58c52c92a4d4](https://www.kubii.com/fr/cartes-micro-controleurs/3091-carte-microbit-bbc-v2-5051259252585.html?mot_tcid=1436612e-e738-4468-b49f-58c52c92a4d4))
- **Câbles micro-USB** : pour alimenter et programmer le micro:bit
- **Batterie externe (en option)** : Pour un fonctionnement portable si la micro:bit doit être détaché - Vous pouvez trouver le boîtier de piles officielle de micro:bit disponible à l'achat pour environ 2,20 EUR par pack ([ici](https://www.kubii.com/fr/alimentations/4237-1913-support-de-pile-officiel-pour-microbit-3272496317253.html?mot_tcid=693572de-fca1-4287-bbd1-df4c014e258b#/interrupteur-sans) - [https://www.kubii.com/fr/alimentations/4237-1913-support-de-pile-officiel-pour-microbit-3272496317253.html?mot\\_tcid=693572de-fca1-4287-bbd1-df4c014e258b#/interrupteur-sans](https://www.kubii.com/fr/alimentations/4237-1913-support-de-pile-officiel-pour-microbit-3272496317253.html?mot_tcid=693572de-fca1-4287-bbd1-df4c014e258b#/interrupteur-sans))

Diverses modalités de collecte peuvent être organisées :

- **Option 1 (plusieurs cartes)** : Utilisez 5 cartes Micro:bit pour collecter des données simultanément chez 5 élèves pendant une semaine, puis répétez l'opération chez 5 autres élèves la semaine suivante.
- **Option 2 (une seule carte - moins onéreux)** : Utilisez une seule carte Micro:bit et faites-la tourner entre les élèves. Collectez des données pendant 2 jours chez chaque élève, sur une période totale de 15 jours, pour obtenir une représentation de plusieurs points de collecte.

Vous pouvez acheter le kit Micro:bit V2 comprenant le câble USB et le boîtier de piles pour 21 EUR par kit ([ici](https://www.kubii.com/fr/kits-micro-controleurs/3092-kit-microbit-go-v2-5051259252592.html?mot_tcid=e92c2317-81d6-4102-8e90-e56faeb2fe68) - [https://www.kubii.com/fr/kits-micro-controleurs/3092-kit-microbit-go-v2-5051259252592.html?mot\\_tcid=e92c2317-81d6-4102-8e90-e56faeb2fe68](https://www.kubii.com/fr/kits-micro-controleurs/3092-kit-microbit-go-v2-5051259252592.html?mot_tcid=e92c2317-81d6-4102-8e90-e56faeb2fe68)), ou 177 EUR pour 10 kits ([ici](https://www.kubii.com/fr/kits-micro-controleurs/3093-kit-microbit-club-v2-5051259252615.html?mot_tcid=97a4ea0c-3489-461e-ad35-4aec28defa2d) - [https://www.kubii.com/fr/kits-micro-controleurs/3093-kit-microbit-club-v2-5051259252615.html?mot\\_tcid=97a4ea0c-3489-461e-ad35-4aec28defa2d](https://www.kubii.com/fr/kits-micro-controleurs/3093-kit-microbit-club-v2-5051259252615.html?mot_tcid=97a4ea0c-3489-461e-ad35-4aec28defa2d)).

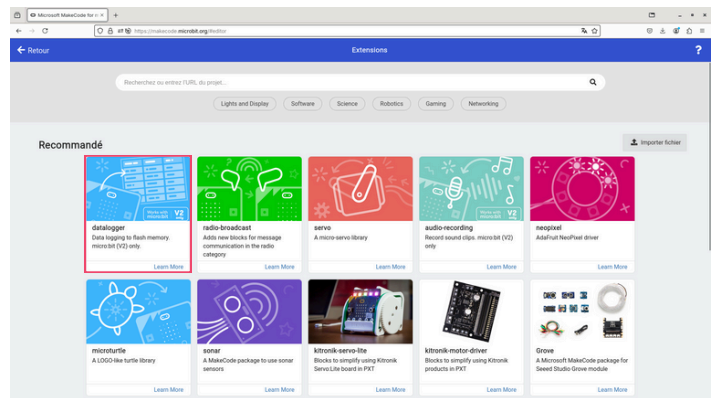
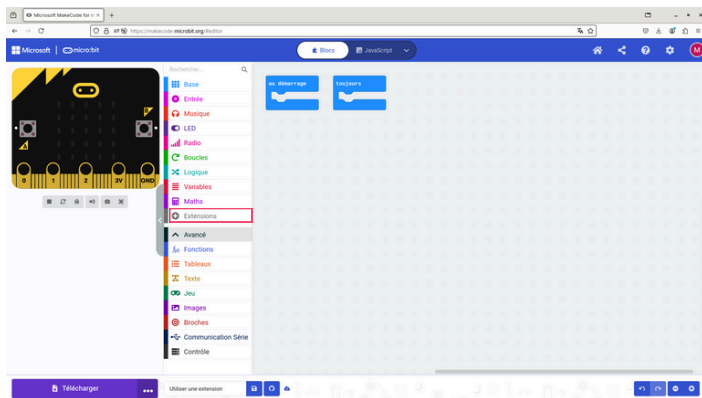
- **Ordinateur ou tablette** : pour écrire et télécharger du code.
- **Environnement de programmation** : [Éditeur en ligne MakeCode](https://makecode.microbit.org/#editor) - <https://makecode.microbit.org/#editor>

## Câblage et utilisation d'une carte Micro:bit

**Étape 1 : Programmation de la carte Micro:bit. Connectez la carte Micro:bit** : Connectez la carte micro:bit à l'ordinateur sur lequel vous avez réalisé le programme grâce à l'éditeur MakeCode. Une fois connectée, la carte micro:bit apparaîtra sur l'ordinateur comme un disque amovible (ex. : "MICROBIT").

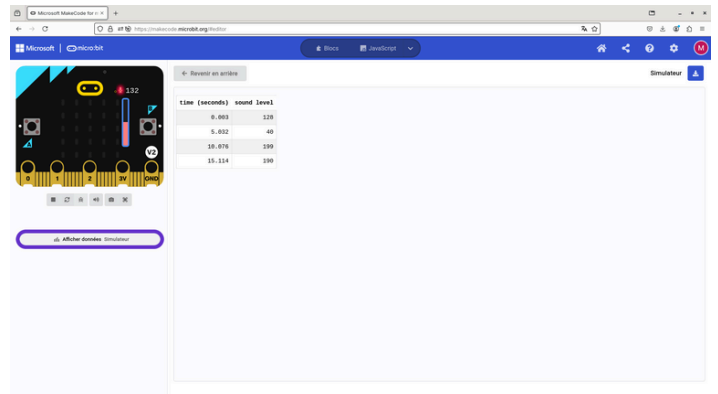
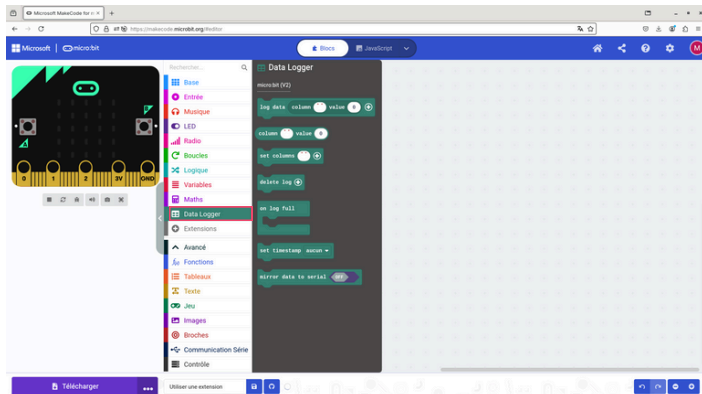
**Écrire le programme** : Ouvrez l'éditeur MakeCode pour créer un programme qui collecte des données de bruit à l'aide des capteurs intégrés de la carte de programmation Micro:bit V2. Donnez un nom clair à votre projet avant de commencer.

Une fois sur l'éditeur et après avoir créé votre nouveau projet, vous obtiendrez l'écran par défaut « prêt à l'emploi » et vous devrez installer une **extension**. Les **extensions** dans MakeCode sont des groupes de blocs de code qui ne sont pas directement inclus dans les blocs de code de base de MakeCode. Les extensions, comme leur nom l'indique, ajoutent des blocs pour des fonctionnalités spécifiques. Il existe des **extensions** pour un large éventail de fonctionnalités très utiles, ajoutant des capacités de manette de jeu, de clavier, de souris, de servomoteur et de robotique et bien plus encore. Dans les colonnes d'affichage des blocs, cliquez sur le bouton **EXTENSIONS**. Dans la liste des extensions disponibles, recherchez l'**extension Datalogger** qui sera utilisée pour cette activité. Cliquez sur l'extension que vous souhaitez utiliser et un nouveau groupe de blocs apparaîtra sur l'écran principal.



1 Ouvrez le menu des extensions depuis l'éditeur MakeCode micro:bit.

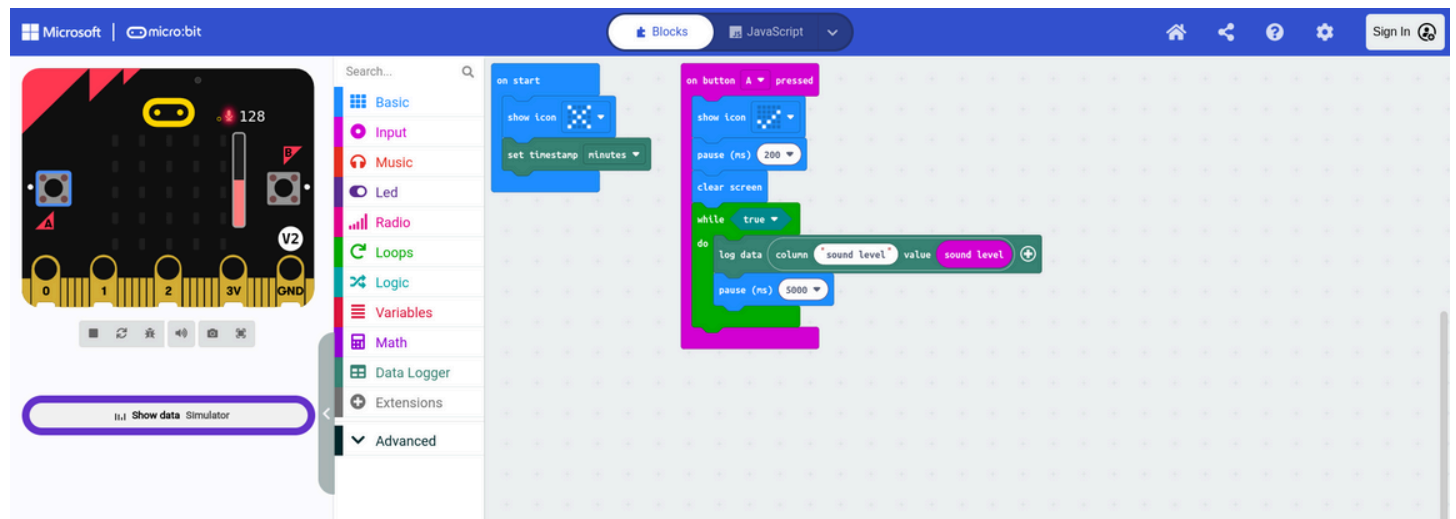
2 Sélectionnez l'extension datalogger dans la liste (vous pouvez également utiliser l'outil de recherche).



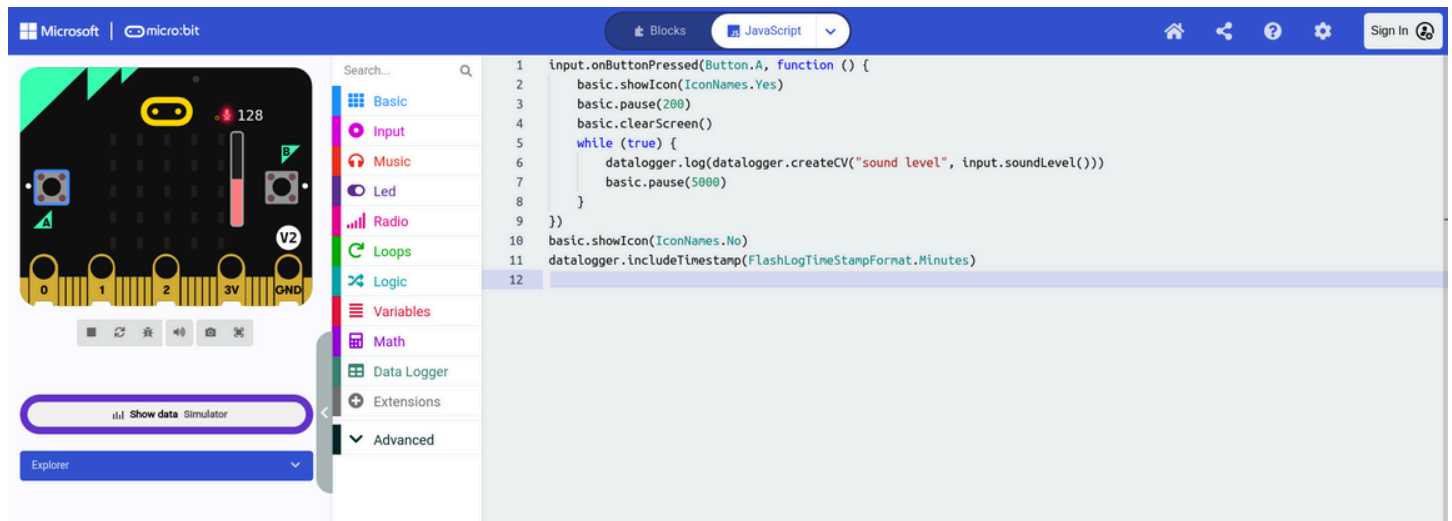
3 L'extension datalogger apparaît dans la liste des blocs disponibles depuis votre éditeur.

4 Le datalogger vous permet d'accéder à un simulateur d'enregistrement de données.

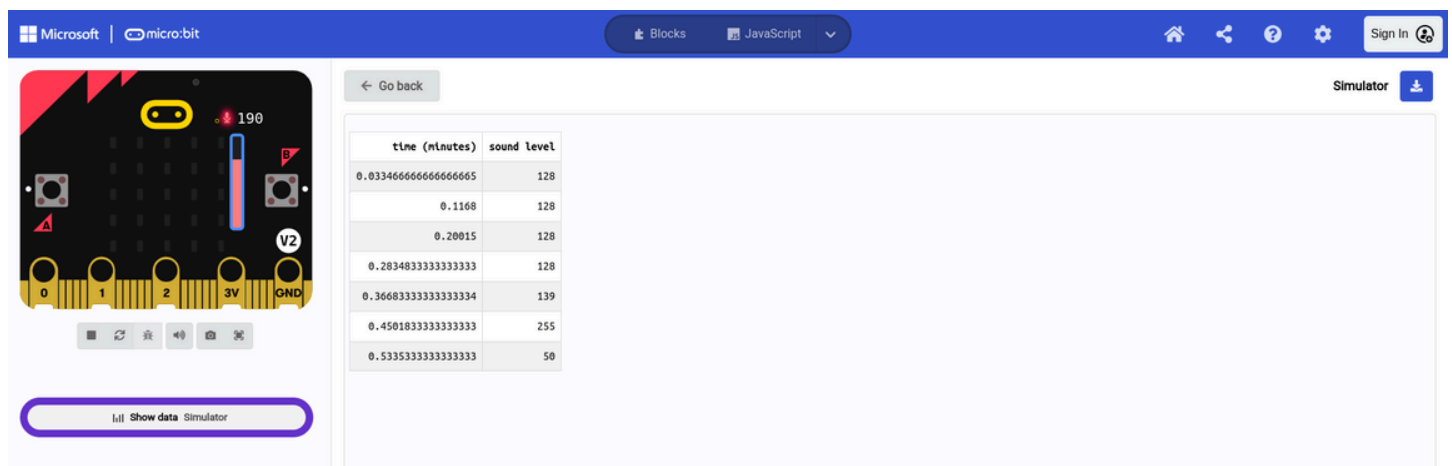
Ensuite, vous pouvez commencer à organiser vos blocs en suivant le code fourni ci-dessous (ajouter une boucle infinie, enregistrer les données dans le datalogger...).



Vous pouvez également copier-coller le code dans l'éditeur Javascript.



Testez le programme en utilisant le simulateur de MakeCode.



Une fois votre programme fonctionnant correctement sur le simulateur, transférez-le sur votre Micro:bit : cliquez sur « **Télécharger** » dans MakeCode pour générer un fichier .hex. Ce fichier contient le programme compilé qui permettra à la carte de fonctionner. Copiez le fichier .hex de votre dossier de téléchargement sur le lecteur amovible « **MICROBIT** ». Une fois le fichier copié, la carte redémarre automatiquement et exécute le code.

**Étape 2 : Placer la Micro:bit et commencer à enregistrer les données.** Une fois programmé, placez la micro:bit pour collecter les données dont vous avez besoin, par exemple dans le jardin d'un élève, dans le parc à côté de l'école, dans la mairie... selon votre choix d'emplacement. Utilisez une batterie externe pour vous assurer que votre carte collectera des données au cours de la semaine par exemple.

**Une fois positionné, appuyez sur le bouton « A » du MicroBit pour démarrer l'enregistrement des données via le programme.**

**Étape 3 : Récupération des données et préparation de la carte pour la prochaine session d'enregistrement.** Une fois la période de collecte terminée, vous pouvez récupérer vos données à partir du fichier appelé « **MY\_DATA.HTM** », disponible sur le lecteur micro:bit. Copiez-le sur votre ordinateur et renommez-le avec la date du jour (par exemple, LOCATIONNAME\_YYYY\_MM\_DD.HTM).

Après avoir copié et renommé le fichier, **supprimez le fichier MY\_DATA.HTM de la MicroBit** pour libérer de l'espace et permettre un nouvel enregistrement de données.

Une fois ouvert, les fichiers de données seront accessibles au format HTML. Ils fourniront toutes les données collectées et vous permettront de les télécharger au format .csv.

## Utiliser et comprendre le code

Voici le code Javascript utilisé pour programmer une carte micro:bit afin de collecter régulièrement des données de bruit :



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

**Fonctionnement du code.** Ce programme mesure le niveau sonore ambiant (en décibels) toutes les 5 secondes (l'intervalle peut être modifié pour correspondre à 1 minute, 5 minutes, 2 fois par heure ...) et compile les informations dans un **“datalogger”** depuis lequel nous pouvons télécharger un fichier .csv.



Un fichier **.csv** (Comma-Separated Values, ou valeurs séparées par des virgules) est un format de fichier texte utilisé pour stocker des données tabulaires (comme dans un tableau ou une feuille de calcul). Chaque ligne du fichier représente une ligne de données, et chaque valeur dans une ligne est séparée par un délimiteur (souvent une virgule, mais parfois un point-virgule ou une tabulation). Il est possible de récupérer les données d'un fichier .csv dans un tableur type Excel ou LibreOffice Calc. Dans Excel, ouvrez le logiciel, cliquez sur **Fichier > Ouvrir**, sélectionnez le fichier .csv, et configurez les délimiteurs si nécessaire via l'outil d'importation. Dans LibreOffice Calc, suivez un processus similaire : cliquez sur **Fichier > Ouvrir**, sélectionnez le fichier, et utilisez l'assistant d'importation pour choisir le délimiteur (par exemple, virgule ou point-virgule). Dans les deux cas, les données s'affichent sous forme de tableau, prêtes à être analysées.

**Initialisation de l'événement d'appui sur le bouton « A »:** Lorsque l'utilisateur appuie sur le **bouton « A »** du MicroBit, la fonction `input.onButtonPressed(Button.A, function () {...})` est déclenchée.

**Affichage de l'icône "Yes" pendant l'exécution:** Avant de démarrer l'enregistrement des données, le programme affiche l'icône « Yes » (`basic.showIcon(IconNames.Yes)`) pendant **200 millisecondes** (0,2 seconde) pour indiquer que le processus d'enregistrement a démarré.

**Pause de 200 millisecondes:** Après avoir affiché l'icône « Oui », le programme attend **200 millisecondes** en utilisant `basic.pause(200)`.

**Effacer l'écran:** Après la pause de 200 millisecondes, l'écran est effacé avec `basic.clearScreen()`, qui prépare l'écran pour ce qui suit sans être encombré d'images.

**Boucle infinie de collecte de données:** Le programme entre dans une boucle infinie `while (true)`. Cela signifie que les données seront collectées et enregistrées sans fin jusqu'à ce que la MicroBit soit éteint ou redémarré.

**Enregistrement des données dans le datalogger:** À chaque itération de boucle, le programme enregistre les valeurs des capteurs MicroBit concernant le **niveau sonore** à l'aide de `input.soundLevel()`, qui capture le niveau sonore ambiant.

Le capteur mesure une valeur relative et ne fournit pas directement des valeurs en unités standard comme les décibels (dB). Plus précisément, le capteur mesure l'intensité perçue. Cette valeur est une estimation numérique (de 0 à 255), où 0 représente la valeur minimale (silence complet/obscurité totale) et 255 la valeur maximale (bruit très fort/lumière intense).

Ces valeurs sont enregistrées dans le **datalogger** sous forme de variables nommées « niveau sonore ». Cela se fait via la fonction `datalogger.log()` :

```
datalogger.log(  
    datalogger.createCV("niveau sonore", input.soundLevel())  
)
```

La fonction `createCV` permet de créer un « CV » (valeur de contexte) pour chaque capteur, et la fonction `datalogger.log` permet d'enregistrer ces valeurs dans un fichier sur la MicroBit.

**Pause de 5 000 millisecondes avant la prochaine lecture:** Après chaque enregistrement, le programme attend **5 000 millisecondes** (5 secondes) avant de relire les valeurs du capteur. Ceci est réalisé avec `basic.pause(5000)`. Vous pouvez modifier la durée de la pause pour capturer plus ou moins de données (par exemple, toutes les minutes).

**Horodatage des données (inclus via `datalogger.includeTimestamp`):** En dehors de la fonction liée au bouton, la commande `datalogger.includeTimestamp(FlashLogTimeStampFormat.Minutes)` est utilisée pour inclure un horodatage avec chaque enregistrement de données. Le format d'horodatage est en minutes, ce qui signifie que chaque enregistrement aura un indicateur de temps basé sur les minutes écoulées depuis le démarrage du programme.

**Affichage de l'icône "No" avant l'exécution:** Avant que l'utilisateur n'appuie sur le bouton « A », le programme affiche une icône « **No** » (`basic.showIcon(IconNames.No)`) pour indiquer que la MicroBit attend l'action de l'utilisateur.