Fiche - Programmation & Capteurs

Mesurer les données environnementales : lumière, bruit, température

Matériel et outils nécessaires

Pour programmer une carte micro:bit pour collecter l'humidité et la température, vous aurez besoin du matériel suivant :

- **Carte Micro:bit V2 et ses capteurs intégrés :** La carte programmable principale comprenant un capteur de lumière intégré via son écran LED, un capteur de niveau sonore intégré et un capteur de température intégré - Environ 19 EUR par micro:bit (Voir les prix)
- **Câble micro-USB** : Pour alimenter et programmer la micro:bit
- **Batterie externe** (facultatif): Pour un fonctionnement portable si la micro:bit doit être détaché - Vous trouverez le boitier de piles officielle micro:bit disponible à l'achat pour environ 2,20 EUR par bloc ici

Vous pouvez également acheter le kit Micro:bit V2 comprenant le câble USB et le boitier de piles pour 21 EUR par kit(ici) ou 177 EUR pour 10 kits (ici)

- **Ordinateur ou tablette**: Pour écrire et télécharger du code.
- **Environnement de programmation** : Editeur MakeCode en ligne

Nous recommandons pour cette étape de programmer au moins 3 à 6 cartes micro:bit pour les partager entre les élèves et recueillir plus d'informations et de données. Vous pouvez le faire avec une seule carte, mais vous devrez soit étendre la durée globale de la période de collecte, soit réduire la durée de la période de collecte par élève de 7 à 3 jours.

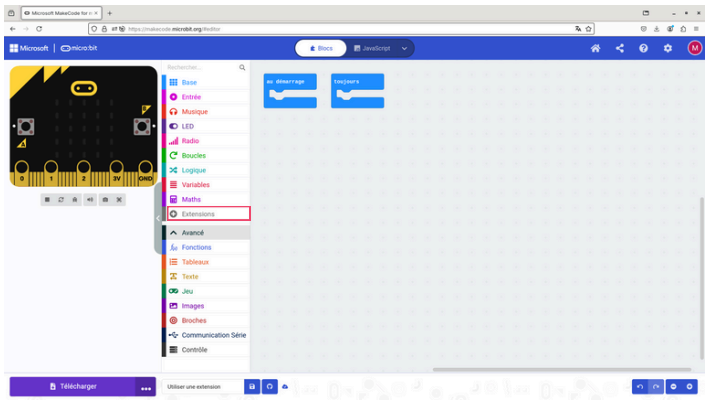
Câblage et utilisation d'une carte Micro:bit

Suivez ces étapes pour programmer, placer, enregistrer et récupérer des données environnementales à l'aide du micro:bit.

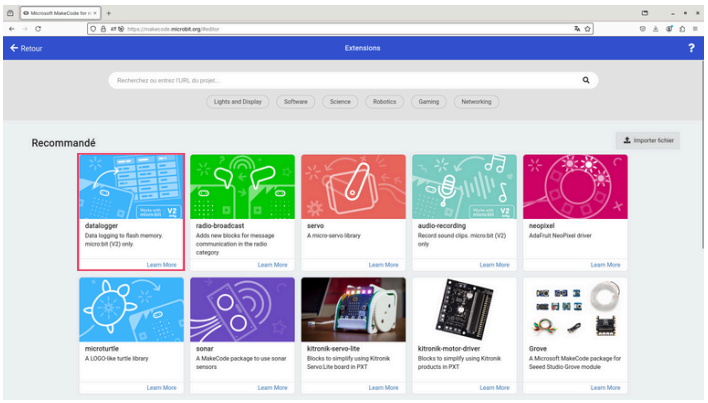
Étape 1 - Programmer la carte Micro:bit

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").

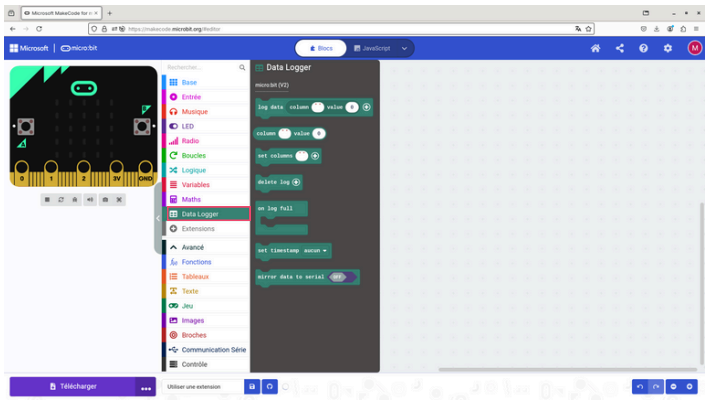
Écrire le programme : Ouvrez l'éditeur MakeCode pour créer un programme qui collecte des données de bruit à l'aide des capteurs intégrés de la carte de programmation Micro:bit V2. Donnez un nom clair à votre projet avant de commencer. Une fois dans l'éditeur, après avoir créé votre nouveau projet, vous accéderez à 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.



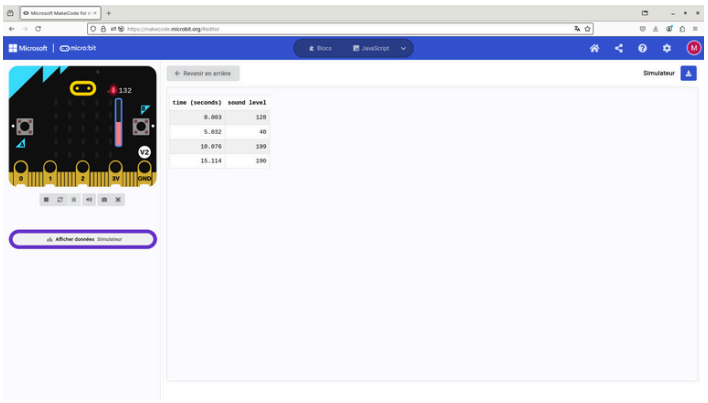
1 **Ouvrez le menu des extensions depuis l'éditeur MakeCode micro:bit.**



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

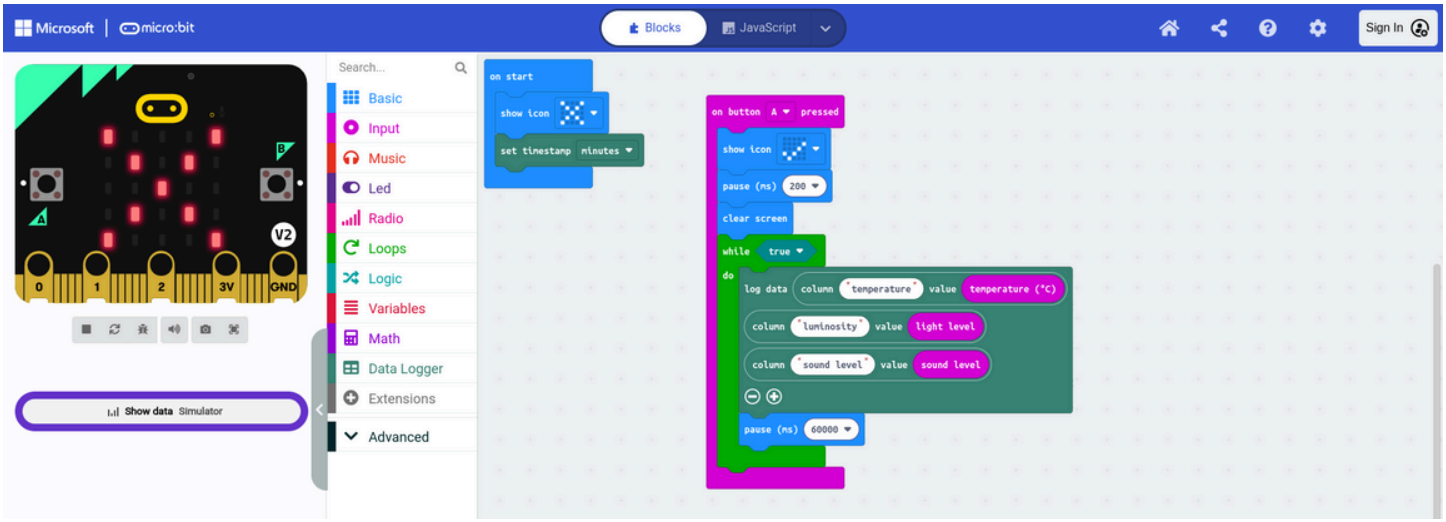


3 **L'extension datalogger apparaît dans la liste des blocs disponibles depuis votre éditeur.**

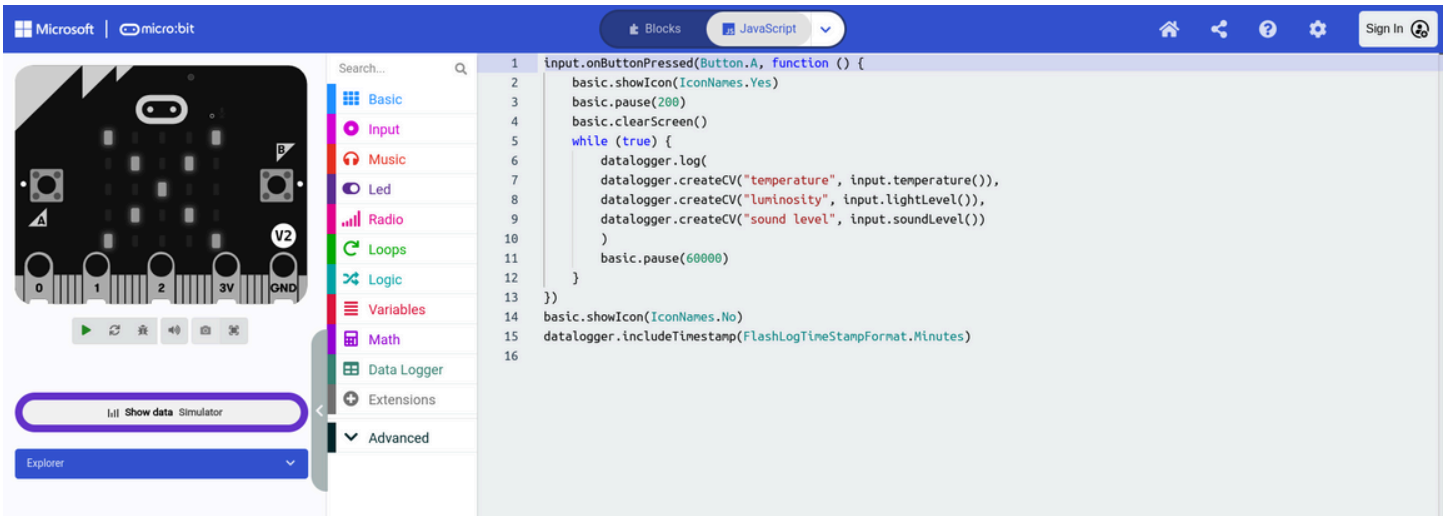


4 **Le datalogger vous permet d'accéder à un simulateur d'enregistrement de données.**

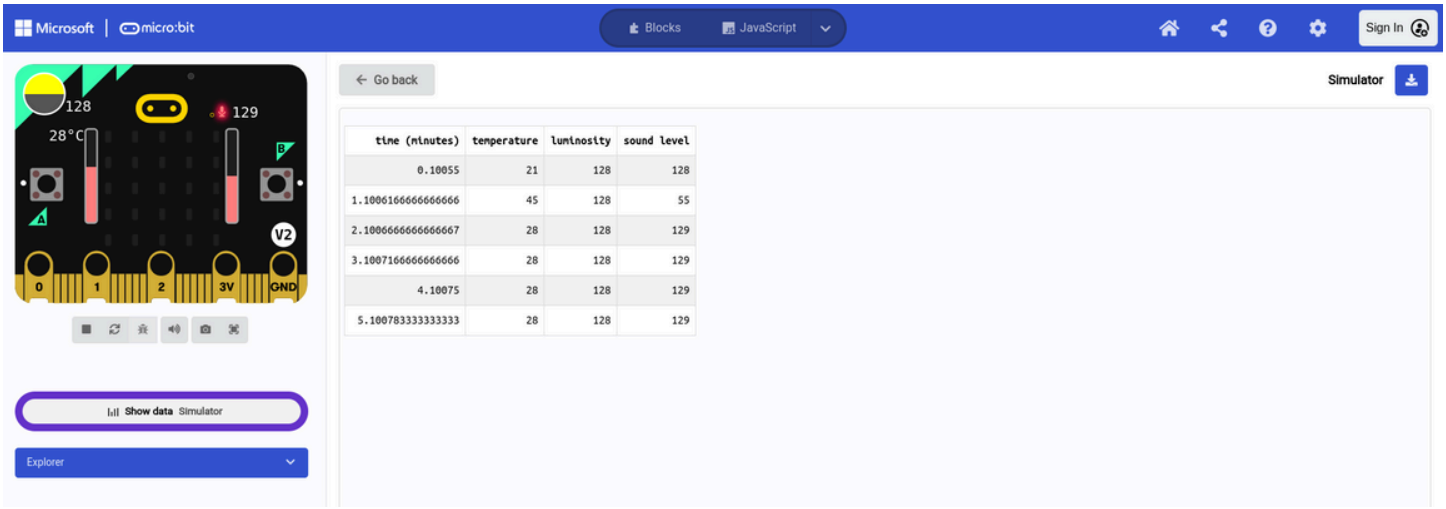
Ensuite, vous pouvez commencer à organiser vos blocs en suivant le code fourni ci-dessous (ajouter une boucle infinie, enregistrer les données dans le datalogger...).



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 le micro:bit pour collecter les données dont vous avez besoin, c'est-à-dire près de votre lit dans une zone où il peut enregistrer avec précision la lumière, le bruit et la température sans obstruction. **Utilisez un ordinateur ou une batterie externe pour alimenter le micro:bit en continu pendant l'enregistrement**. Assurez-vous que chaque nuit, la carte est à nouveau placée exactement dans la même position pour enregistrer des données comparables.

Avant d'aller vous coucher, 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

Chaque matin, pour éviter toute perte de données, nous vous recommandons de débrancher le micro:bit de sa source d'alimentation pour stopper l'enregistrement des données et de **connecter la micro:bit à votre ordinateur pour accéder au fichier compilé pendant la nuit par le datalogger (qui s'appellera « MY_DATA.HTM, disponible sur le lecteur micro:bit).**

Copiez ce fichier sur votre ordinateur et renommez-le avec la date du jour (par exemple, BOARD1_NAME_YYYY-MM-DD.HTM).

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

Répétez le processus pour la séance suivante, c'est-à-dire le lendemain soir avant d'aller vous coucher.

A la fin de la période de collecte, vous pourrez récupérer l'ensemble des fichiers collectés sur les différentes cartes micro:bit. 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 sur la lumière, le bruit et la température :

```
input.onButtonPressed(Button.A, function () {
    basic.showIcon(IconNames.Yes)
    basic.pause(200)
    basic.clearScreen()
    while (true) {
        datalogger.log(
        datalogger.createCV("temperature", input.temperature()),
        datalogger.createCV("luminosite", input.lightLevel()),
        datalogger.createCV("niveau sonore", input.soundLevel())
        )
        basic.pause(60000)
    }
})
basic.showIcon(IconNames.No)
    datalogger.includeTimestamp(FlashLogTimeStampFormat.Minutes)
```

Comment fonctionne le code ? Ce programme mesure le niveau sonore ambiant, la température et la luminosité. Toutes les minutes (l'intervalle peut être modifié pour correspondre à 10 secondes, 5 minutes, 2 fois par heure...) le programme compile les informations dans un "**datalogger**" à partir duquel on peut 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:

- **temperature**: `input.temperature()`, qui récupère la température actuelle en degrés Celsius.
- **luminosite**: `input.lightLevel()`, qui mesure le niveau de lumière ambiante.
- **niveau sonore**: `input.soundLevel()`, qui capte le niveau sonore ambiant.

Le **niveau sonore** et le **niveau lumineux** mesurent une valeur relative et n'ont pas d'unités standard comme les décibels (**dB**) pour le niveau sonore ou les lux (**lx**) pour la luminosité. 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). La **température** est mesurée en degrés Celsius(**°C**). Ces valeurs sont enregistrées dans le **datalogger** sous forme de variables portant des noms respectifs (« temperature », « luminosite », « niveau sonore »). Cela se fait via la fonction `datalogger.log()` :

```
datalogger.log(
    datalogger.createCV("temperature", input.temperature()),
    datalogger.createCV("luminosite", input.lightLevel()),
    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 60 000 millisecondes avant la prochaine lecture : Après chaque enregistrement, le programme attend 60 **000 millisecondes** (1 minute) avant de relire les valeurs du capteur. Ceci est réalisé avec `basic.pause(60000)`. 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.