Fiche - Programmation & Capteurs

Mesure de l'amplitude des mouvements pendant la nuit

Matériel nécessaires, câblage et utilisation d'une carte Micro:bit

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 officiella 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)

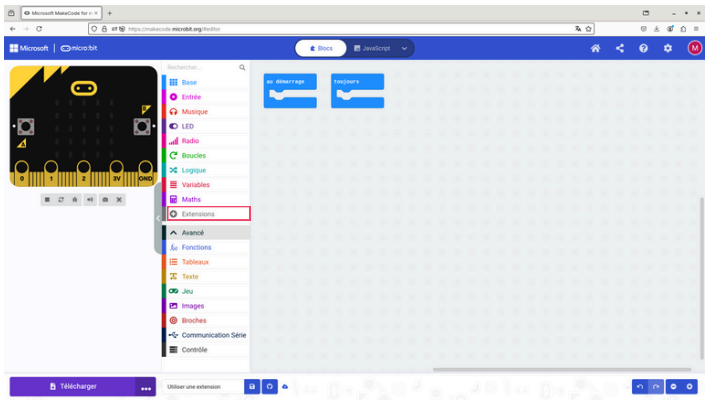
- **Ordinateur ou tablette**: Pour écrire et télécharger du code.
- **Environnement de programmation** : Editeur MakeCode en ligne

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

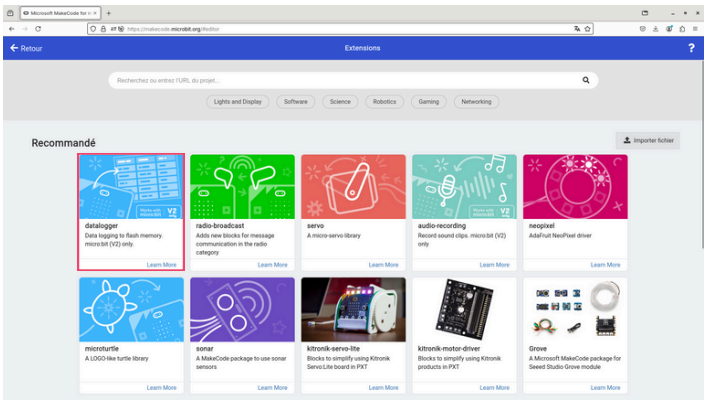
Étape 1 - Programmer la carte Micro:bit

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").

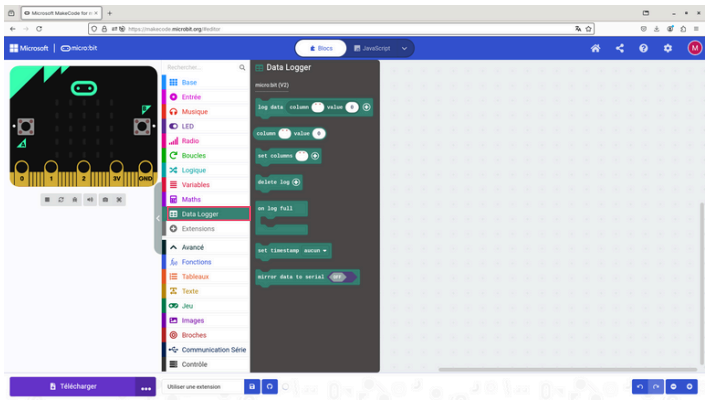
Écrire le programme : Ouvrez l'éditeur MakeCode pour créer un programme qui collecte des données de bruit à l'aide des capteurs intégrés de la carte de programmation Micro:bit V2. Donnez un nom clair à votre projet avant de commencer. Une fois dans l'éditeur, après avoir créé votre nouveau projet, vous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



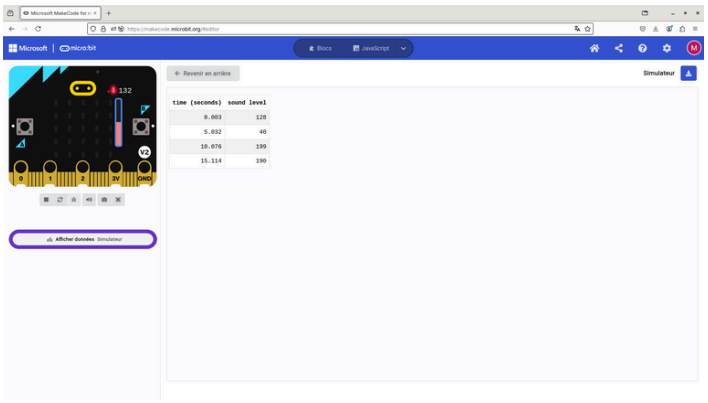
1 **Ouvrez le menu des extensions depuis l'éditeur MakeCode micro:bit.**



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

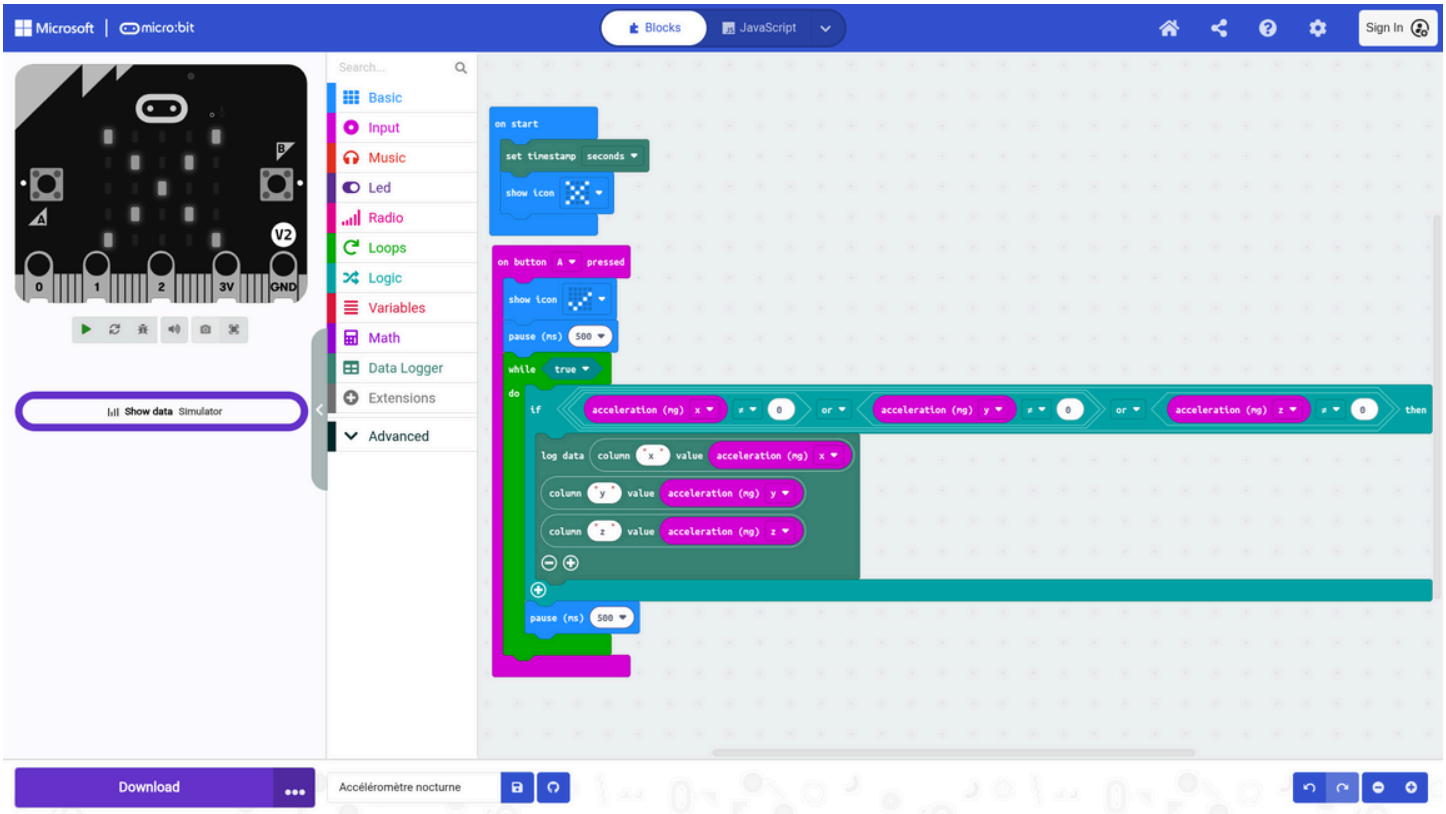


3 **L'extension datalogger apparaît dans la liste des blocs disponibles depuis votre éditeur.**

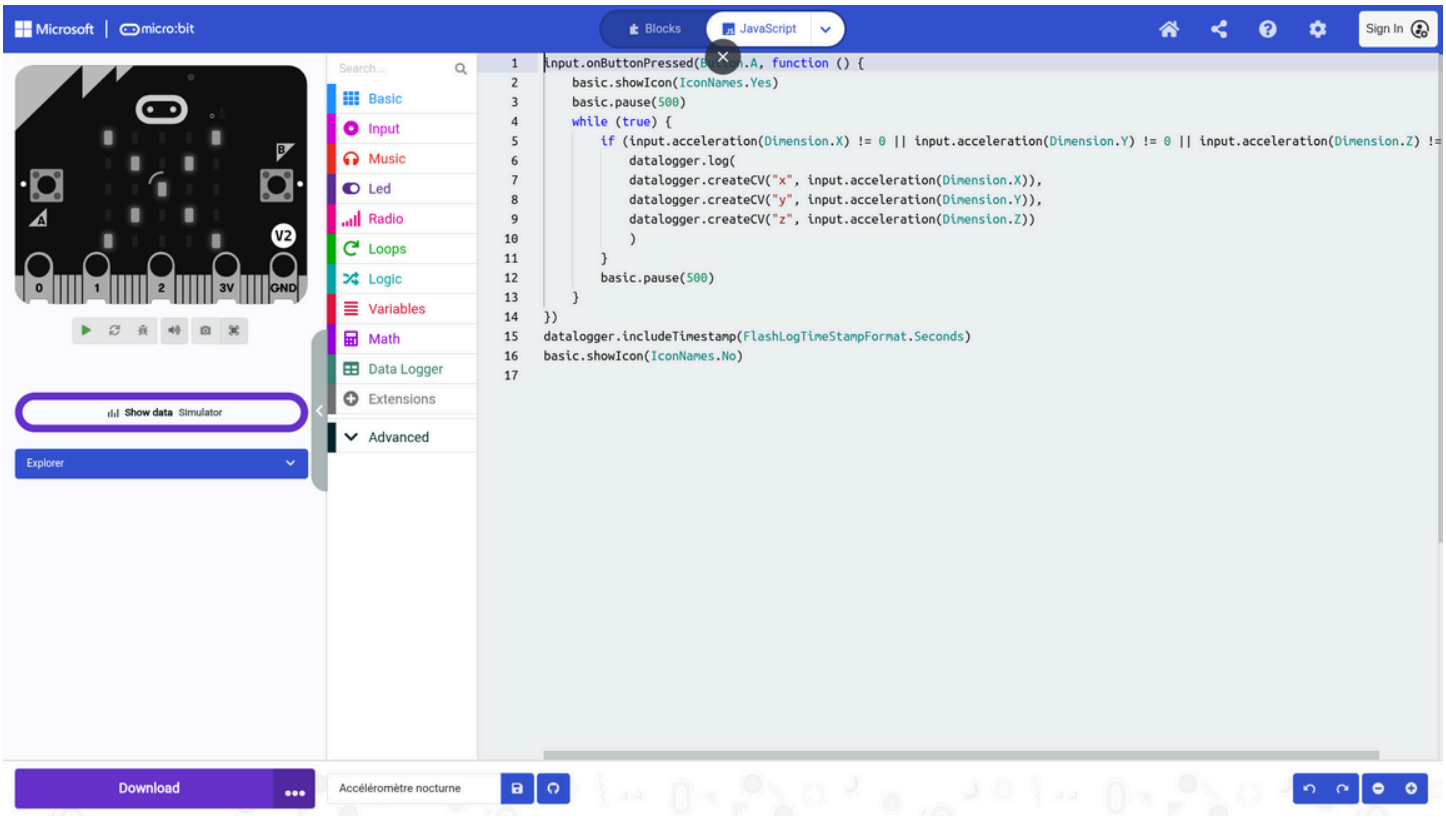


4 **Le datalogger vous permet d'accéder à un simulateur d'enregistrement de données.**

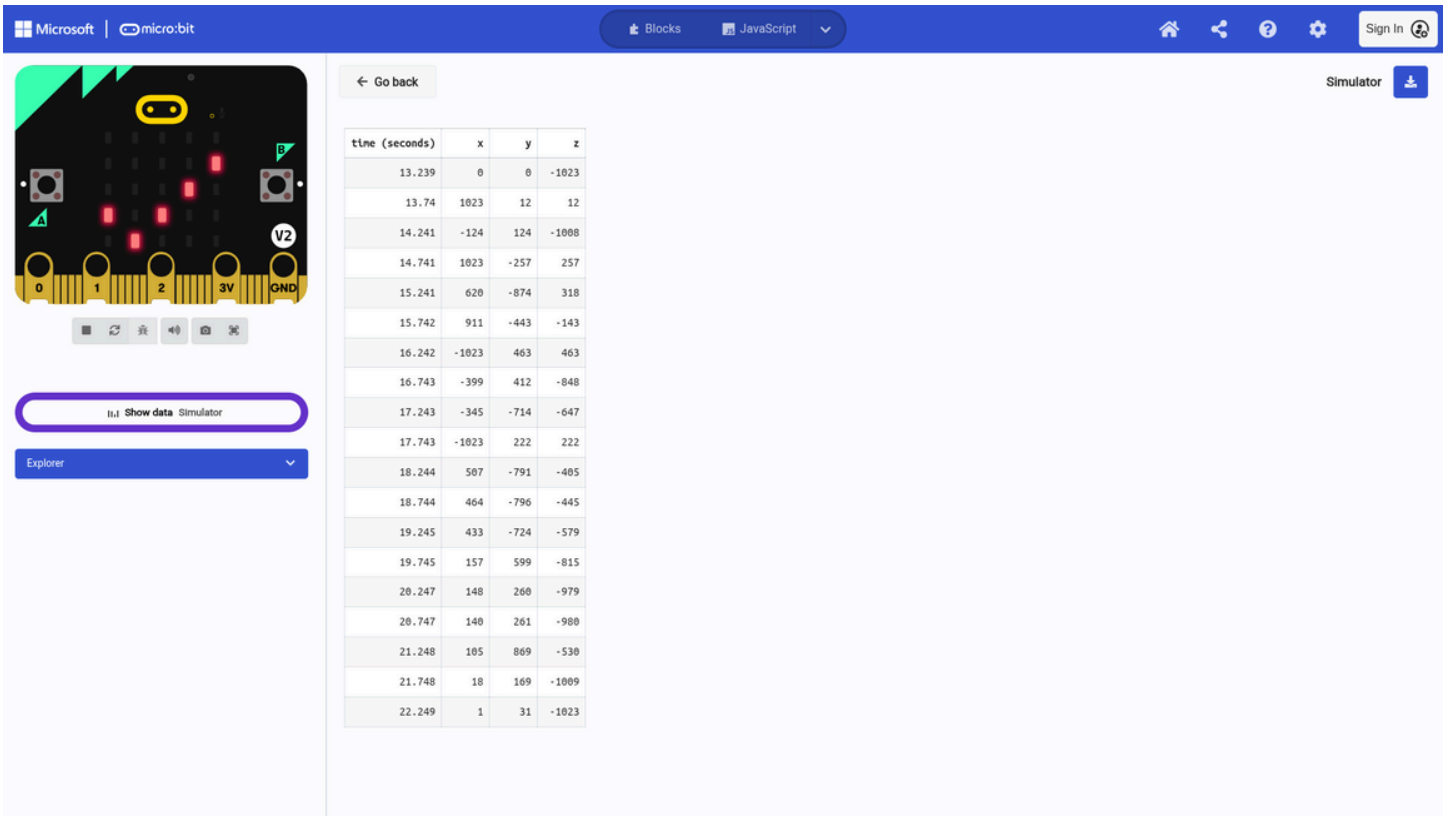
Ensuite, vous pouvez commencer à organiser vos blocs en suivant le code fourni ci-dessous (ajouter une boucle infinie, enregistrer les données dans le datalogger...).



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 la micro:bit pour collecter les données dont vous avez besoin, en utilisant un brassard. Vous trouverez à la fin de cette fiche quelques tutoriels pour créer un brassard micro:bit DIY. **Utilisez une batterie externe pour alimenter la micro:bit en continu pendant l'enregistrement (le kit micro:bit vous donne accès à des boitiers pour piles facilement positionnables sur la carte)**.

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 la 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.

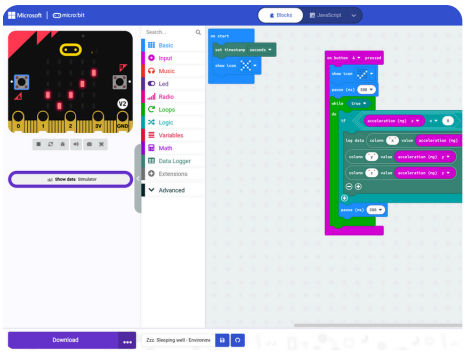
Répétez le processus pour la séance suivante, c'est-à-dire le lendemain soir avant d'aller vous coucher.

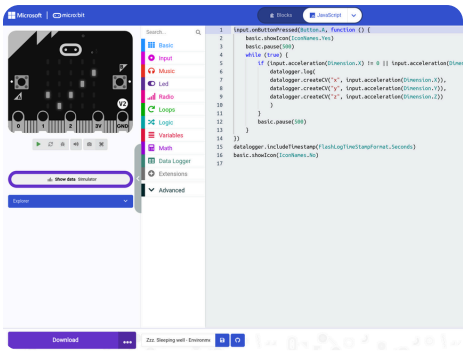
A la fin de la période de collecte, vous pourrez récupérer l'ensemble des fichiers collectés sur les différentes cartes micro:bit. Une fois ouvert, les fichiers de données seront accessibles au format HTML. Ils fourniront toutes les données collectées et vous permettront de les télécharger au format .csv.

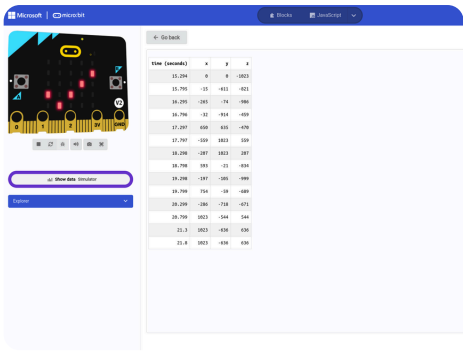
Utiliser et comprendre le code

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

```
input.onButtonPressed(Button.A, function () {
    basic.showIcon(IconNames.Yes)
    basic.pause(500)
    while (true) {
        if (input.acceleration(Dimension.X) != 0 || input.acceleration(Dimension.Y) != 0 ||input.acceleration(Dimension.Z) != 0) {
            datalogger.log(
            datalogger.createCV("x", input.acceleration(Dimension.X)),
            datalogger.createCV("y", input.acceleration(Dimension.Y)),
            datalogger.createCV("z", input.acceleration(Dimension.Z))
            )
        }
        basic.pause(500)
    }
})
datalogger.includeTimestamp(FlashLogTimeStampFormat.Seconds)
basic.showIcon(IconNames.No)
```







<u>Comment fonctionne le code ?</u> Ce programme mesure les valeurs d'accélération captées par l'accéléromètre. Toutes les 500 millisecondes (l'intervalle peut être modifié pour correspondre à 10 secondes, 5 minutes, 2 fois par heure...) le programme vérifie si la carte est en mouvement et si c'est le cas, il compile les données 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.

Ce programme est conçu pour enregistrer les données de l'accéléromètre sur un MicroBit lorsque le bouton « A » est enfoncé et les stocker dans un **datalogger** avec des valeurs pour les axes **X**, **Y** et **Z**.

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. Cela empêche l'enregistrement des données dès que la carte est connectée.

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 **500 millisecondes** (0,5 seconde) pour indiquer que le processus d'enregistrement a démarré.

Pause de 500 millisecondes: Après avoir affiché l'icône « Oui », le programme attend **500 millisecondes** en utilisant `basic.pause(500)`.

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.

Vérification des données de l'accéléromètre: À chaque itération, le programme vérifie si l'une des valeurs d'accélération (sur les axes X, Y ou Z) est différente de zéro. Cela se fait avec la condition :

```
if (input.acceleration(Dimension.X) != 0 || input.acceleration(Dimension.Y) != 0 || input.acceleration(Dimension.Z) != 0)
```

Si l'une de ces valeurs est différente de zéro (ce qui signifie qu'un mouvement a été détecté), le programme enregistre ces données dans le **datalogger**.

Enregistrement des données: Les valeurs d'accélération pour les axes X, Y et Z sont enregistrées à l'aide de la fonction `datalogger.log()` : Cette fonction crée un enregistrement des valeurs d'accélération chaque fois que les valeurs sont différentes de zéro, avec un horodatage pour chaque enregistrement. L'horodatage est automatiquement ajouté grâce à la ligne suivante (expliquée plus loin).

```
datalogger.log(
    datalogger.createCV("x", input.acceleration(Dimension.X)),
    datalogger.createCV("y", input.acceleration(Dimension.Y)),
    datalogger.createCV("z", input.acceleration(Dimension.Z))
)
```

Pause de 500 millisecondes avant la prochaine lecture: Après l'enregistrement des données, le programme s'arrête pendant **500 millisecondes** (`basic.pause(500)`) avant de reprendre les lectures des valeurs de l'accéléromètre.

Horodatage des données (inclus via datalogger.includeTimestamp) : En dehors de la fonction liée au bouton, la commande `datalogger.includeTimestamp(FlashLogTimeStampFormat.Seconds)` est utilisée pour inclure un horodatage en secondes pour chaque enregistrement dans le datalogger. Cela signifie que chaque enregistrement sera accompagné du temps écoulé, en secondes, 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.

Les données: L'accéléromètre renvoie des valeurs en **milli-g** pour chaque axe (X, Y, Z). Cela signifie que si l'accéléromètre détecte une accélération de **1000 milli-g**, cela correspond à une accélération de 1G (ou 9,81 m/s^2) sur cet axe.

Rajouter un brassard

Plusieurs ressources peuvent vous permettre de joindre la carte Micro:bit à un brassard ce qui permettra aux élèves de la porter pendant la nuit. Voici quelques tutoriels et accessoires pertinents qui pourront vous aider à réaliser cette étape :

Smart Coding Watch Kit - micro:bit

Duct Tape Watch

BBC micro:bit wrist holder | mattoppenheim

Yahboom Wrist:bit wearable watch kit based on BBC Micro:bit V2/V1.5 board

CHARGE for micro:bit