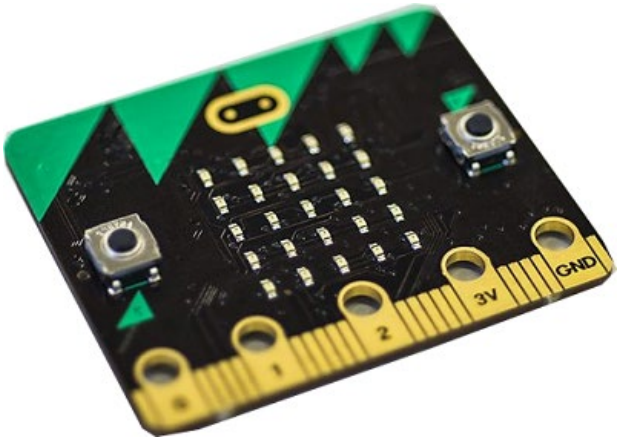
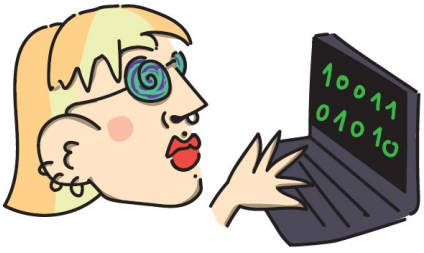


ATELIER PROGRAMMATION MICROBIT V2 – NIV. 1 & NIV. 2

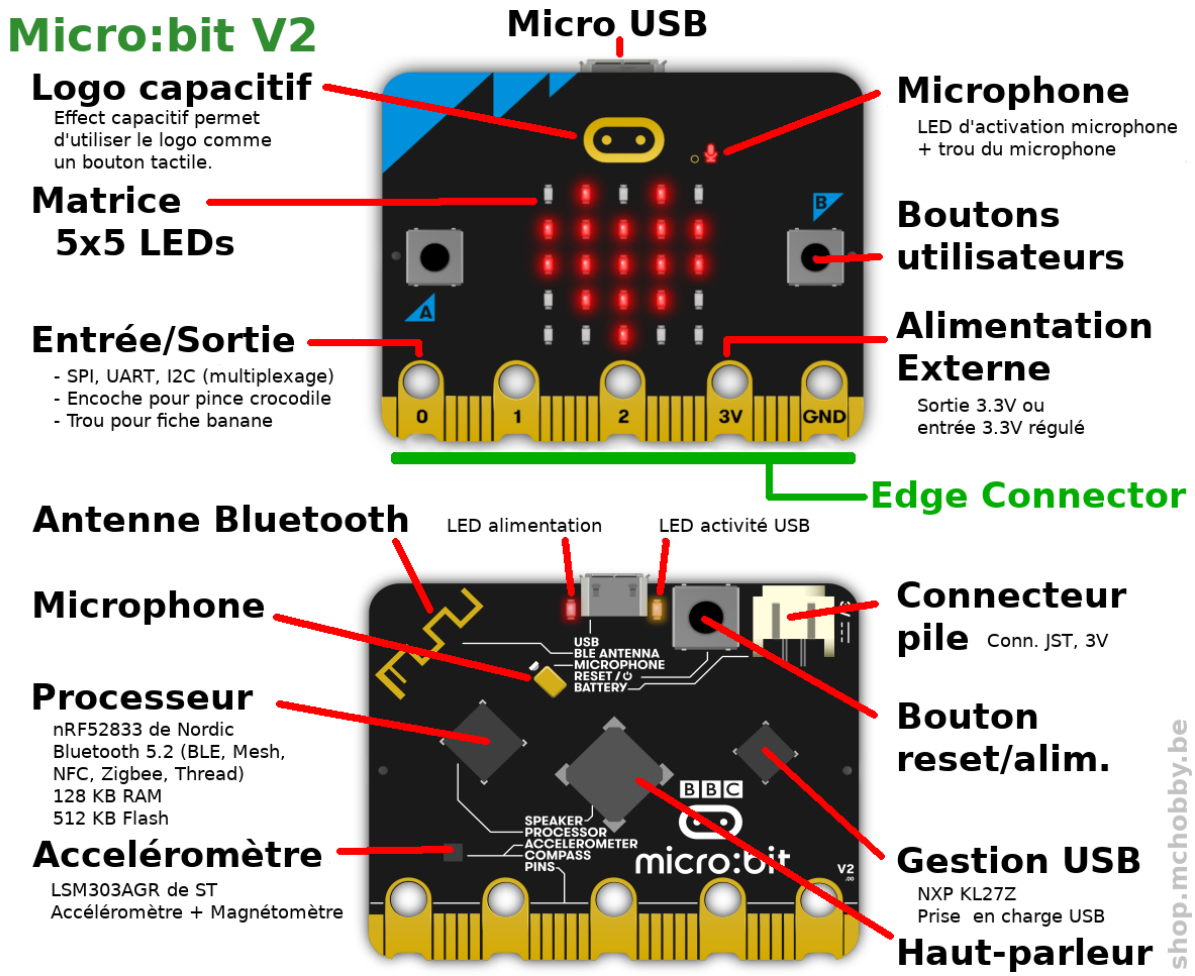
Sommaire :



1. Découverte des composants

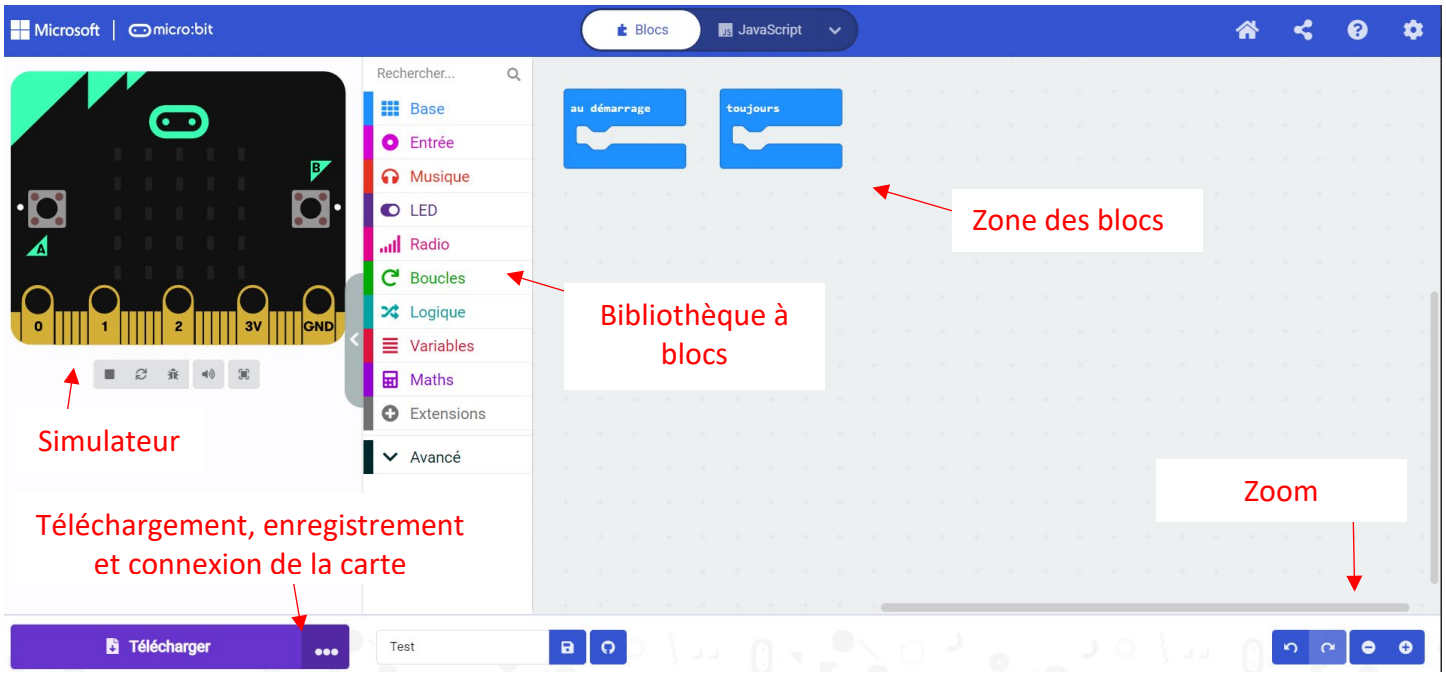
Avant de se lancer à la programmation d'une carte électronique, il est important de connaitre la carte que nous allons utiliser. Cette carte a plein de composants intégrés comme un microphone, une matrice de leds, des boutons poussoirs, un buzzer, … et j'en passe.

Prend un peu de temps pour identifier les composants sur la carte et n'hésite pas à demander des explications tu as du mal à comprendre à quoi ils servent.



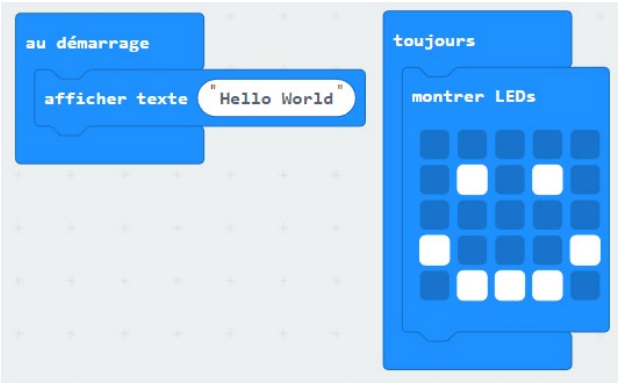
2. Environnement de travail

L'environnement de travail est le lieu où nous allons écrire notre programme. Pour ce faire rendons nous sur internet à cette adresse et lançon un nouveau projet : https://makecode.microbit.org/



3. « Hello World »

Pour commencer, on va afficher sur la matrice de leds un texte défilant puis un smiley. Utilisons le bloc « Démarrage » qui va se lancer qu'une seule fois (contrairement au bloc « Toujours » qui va être une boucle exécutant constamment les blocs qui seront à l'intérieur). Avec le code ci-dessous, affichons le texte Hello World, puis pour Toujours nous allons afficher un smiley. 😊

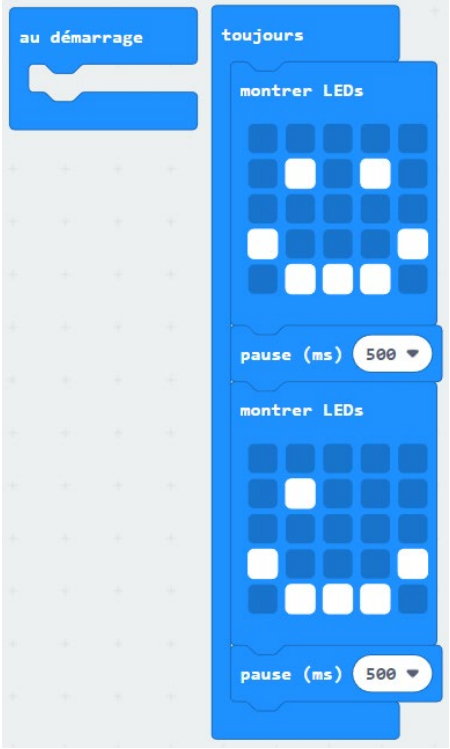


Une autre possibilité est d'utiliser le block « montrer icone ».

Tu l'as compris, les blocs dans Démarrage seront lus qu'une seul fois au Démarrage puis le bloc Toujours sera lu. Au contraire du bloc Démarrage les blocs dans Toujours seront lus indéfiniment à la manière d'une boucle. Les blocs bleus sont dans la bibliothèque de bloc Bases.

Il reste plus qu'à le tester sur le simulateur en appuyant sur la touche « play » .

4. Faire clignoter une image



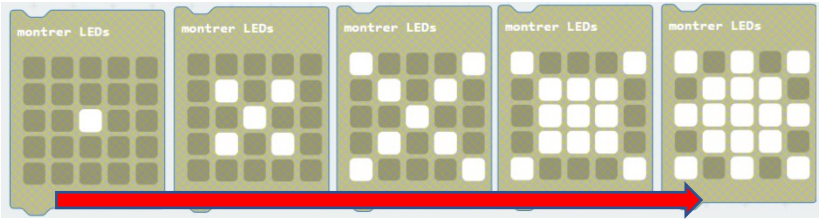
Pour faire clignoter une image nous devons mettre une pause. Avant tout, nous allons supprimer notre ancien travail, et il en sera de même à chaque nouveau chapitre de notre progression.

Ici pour Toujours, faisons clignoter l'œil de notre smiley à un rythme de 0.5s soit 500ms. 😉

Il est important de mettre une pause après la deuxième image.

5. Créer une explosion

Pour donner la sensation d'une explosion, il suffit de mettre une suite d'image avec des pauses courtes entre chacune.

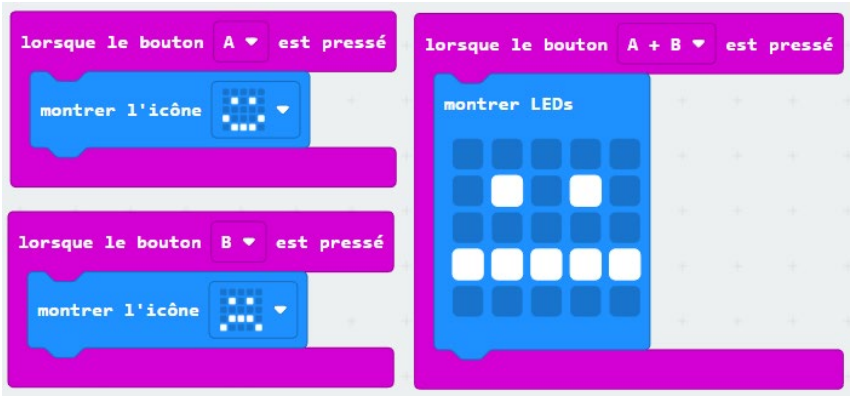


6. Utiliser les boutons A et B

Pour utiliser les boutons, nous devrons aller dans une autre catégorie de blocs, les blocs roses dans Entrée.

Lorsque le bouton A est pressé, nous voulons afficher le smiley 😊 pendant 1 seconde et effacer l'écran.

Lorsque le bouton B est pressé, nous voulons afficher le smiley ☹️ pendant 1 seconde et effacer l'écran.



Lorsque les boutons A+B sont pressés, nous voulons afficher le smiley au sourire neutre pendant 1 seconde et effacer l'écran.

7. Dé numérique et téléversement

Nous allons maintenant réaliser un dé numérique qui se déclenche quand nous appuyons sur le bouton A, et mettre le programme dans notre carte MicroBit.

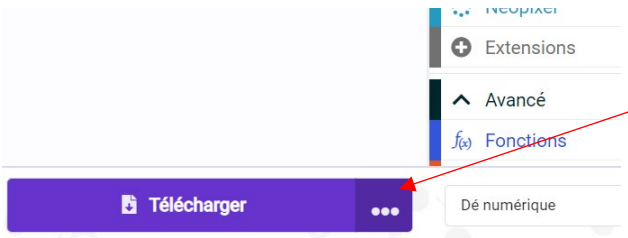
Prenons l'habitude de détailler les étapes que nous souhaitons réaliser, cela correspond à écrire l'algorithme de notre programme.

La première action sera déclencher quand nous allons : *appuyer sur le bouton A → Puis nous voulons montrer/afficher un nombre → ce chiffre sera un chiffre tiré au hasard entre 1 et 6 → attendre 2 secondes puis effacer écran.*

Nous avons besoin d'une nouvelle catégorie pour trouver un chiffre au hasard. Cela se trouve dans Maths.

Maintenant que notre programme est réalisé et que nous l'avons testé sur le simulateur, nous allons mettre le programme dans notre carte. Il faut le téléverser.

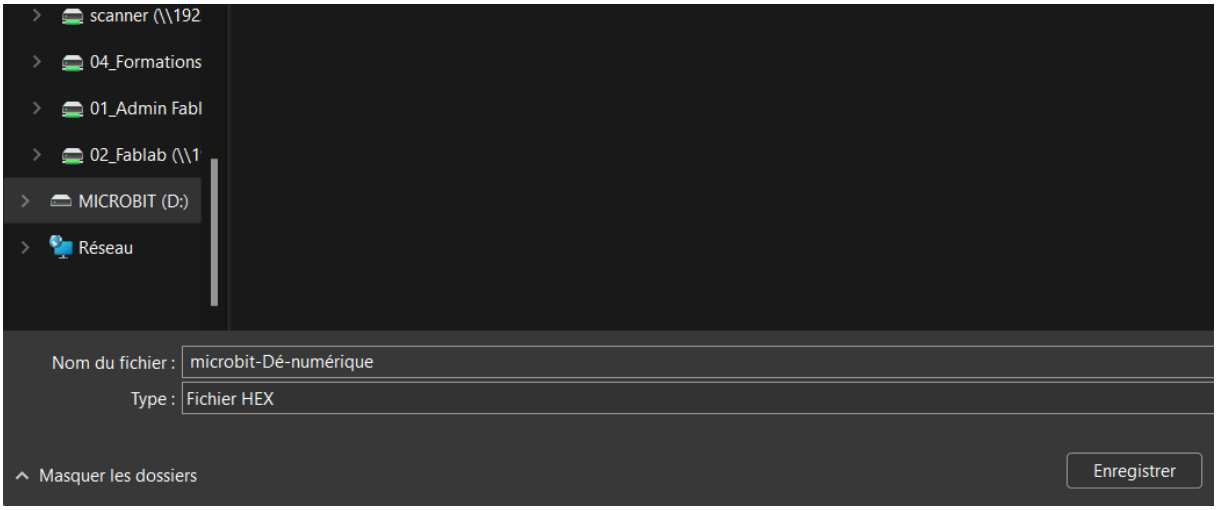
Branchons le câble USB au PC et la carte au port micro USB. **Attention la connectique micro USB de la carte est très fragile. Quand vous la débranchez il faut tirer fort dessus, mais bien droit ! Ne pas faire de mouvement latéral ou de haut en bas !!!**



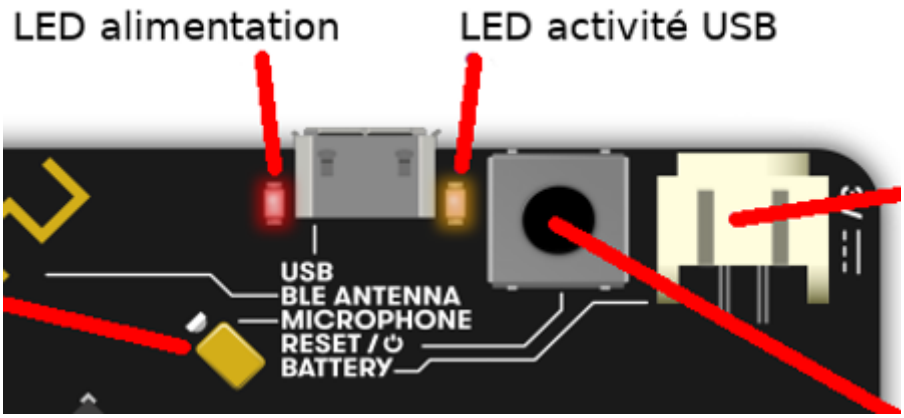
Clicker sur les 3 points

Sélectionner « Télécharger en tant que fichier »

À ce stade, déplacez-vous dans votre explorateur Windows pour enregistrer le fichier dans la carte MicroBit comme s'il s'agissait d'une clé USB.



La LED d'activité USB orange doit se mettre à clignoter durant le téléversement du programme dans la carte.



Une fois terminé, fermez les fenêtres et essayez votre carte tout en la laissant branché. (L'alimentation de la carte passe ici par le port USB)

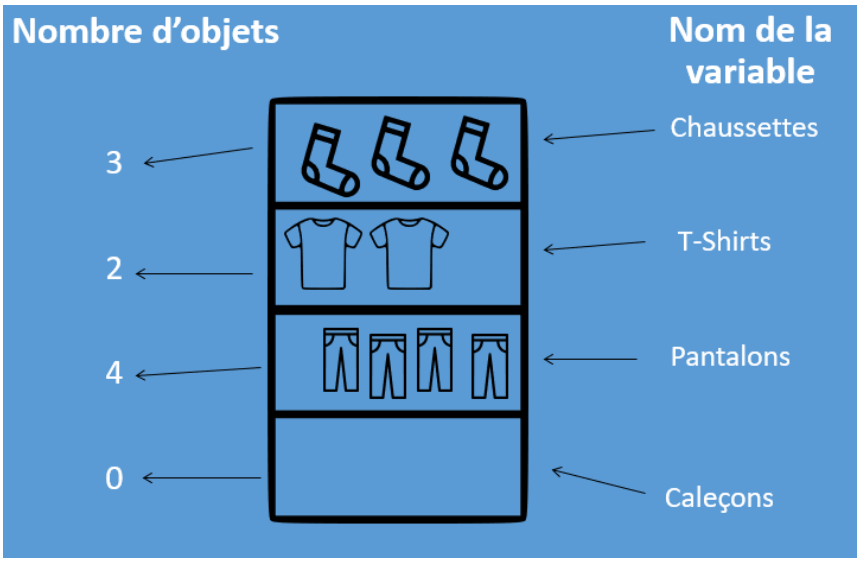
8. Love mètre

Petite variante. Mesurer sa compatibilité amoureuse en pourcentage avec une autre personne. La première personne presse le bouton A pendant que la deuxième presse le bouton B.

Quelle est ta compatibilité avec toi-même ?

😉

9. Introduction aux variables



Une variable est un tiroir dans la mémoire de la carte électronique pour venir stocker un chiffre.

Ici il y a 4 variables. Chaque variable stock un chiffre. On ne peut pas mélanger les Chaussettes avec les T-Shirts.

Dans notre cas, si j'écris Chaussettes = Chaussettes + 2, alors Chaussettes prendra la valeur de 5.

Si après cette opération je fais Chaussettes = Chaussettes – 4, alors Chaussettes aura le chiffre 1 en stock. La notion de variable est fondamentale en programmation.

10. Chifoumi

Grâce à la notion de variables, nous allons pouvoir faire de nouvelles choses. Il suffira d'appeler le nom de cette variable pour pouvoir faire des opérations (Comparaison, condition, addition, soustraction etc…).

Quand on utilise une variable, il est important de définir la valeur de la variable que l'on souhaite au démarrage. Souvent nous mettons la variable à 0. Ici nous allons créer une variable Chifoumi.

L'algorithme est le suivant :

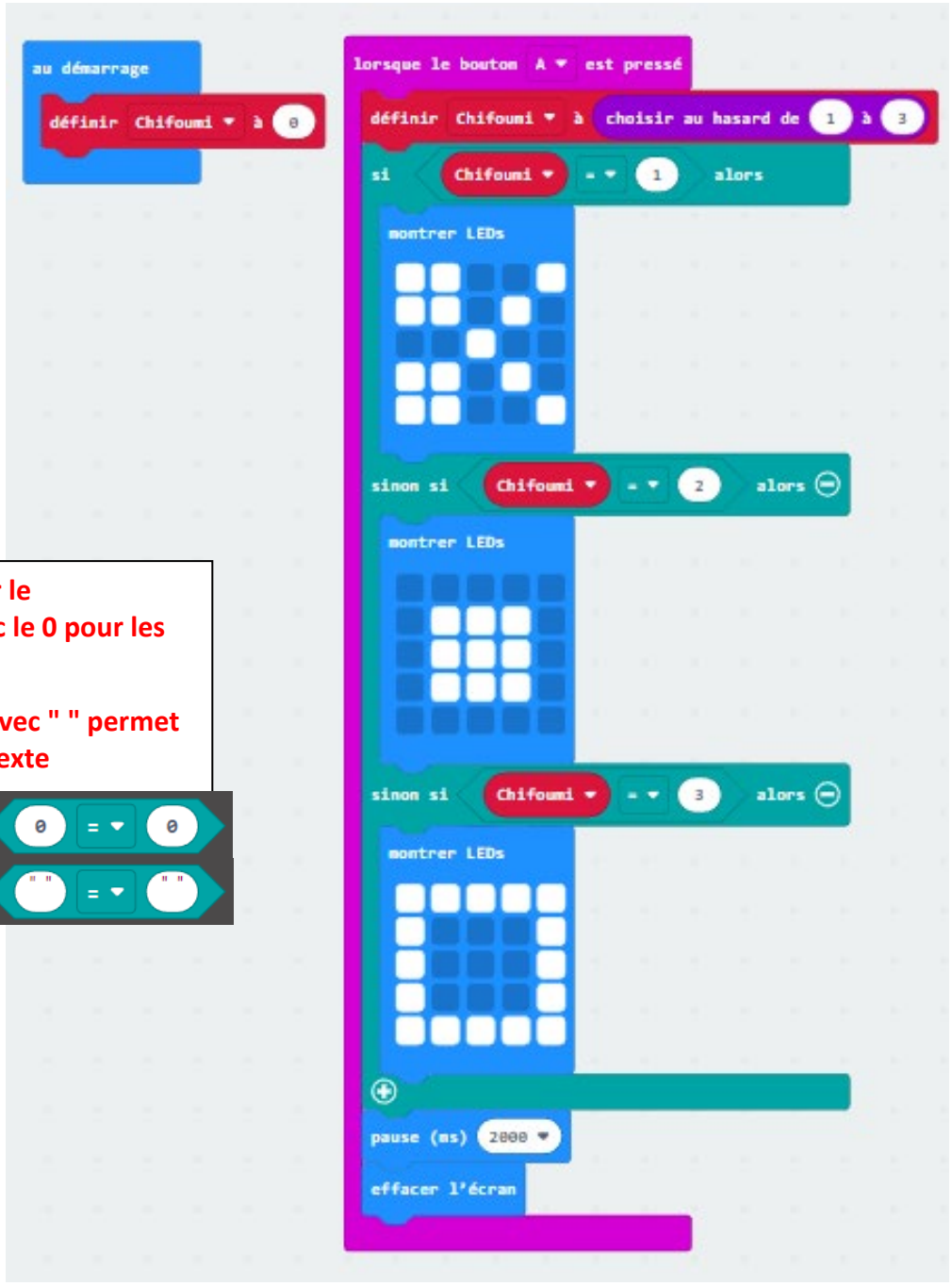
Lorsque je presse le bouton A → je viens mettre/définir dans ma variable un chiffre au hasard entre 1 et 3. (ciseau, pierre, feuille)

→ Si la variable Chifoumi = 1 → Affiche ciseau

→ Si la variable Chifoumi = 2 → Affiche pierre

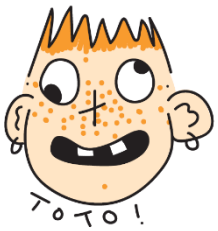
→ Si la variable Chifoumi = 3 → Affiche papier

➔ On attend 2s et on efface l'écran



Attention, utiliser le comparateur avec le 0 pour les chiffres !!

Le comparateur avec " " permet de comparer du texte uniquement.



À vous de jouer maintenant ! Mettez le programme dans votre carte et faites une partie de Chifoumi numérique.

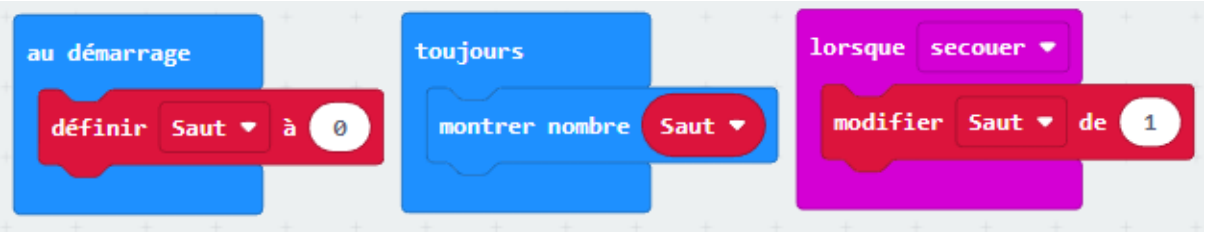
11. Compteur de saut

Nous allons maintenant faire un capteur de saut, dans le but de compter le nombre de saut que nous allons faire.

L'une des possibilités avec une variable, c'est de la modifier plutôt que de la définir. Quand nous définissons, nous imposons la valeur mise dans la variable – ex : Chaussettes = 5

Quand nous modifions on ajoute – ex : modifier Chaussettes de 1, on ajoute 1 à la valeur de Chaussettes.

Ici nous allons sauter en l'air et demander à la carte de compter le nombre de saut que nous allons faire… et donc incrémenter de 1 la variable Saut à chacun de nos sauts.



Tu peux simuler en secouant la carte pour voir si le chiffre s'incrémente.

12. Lampe clap

La lampe s'allume quand on clap dans les mains et reste allumée. Elle s'éteint quand on re-clap dans les mains.

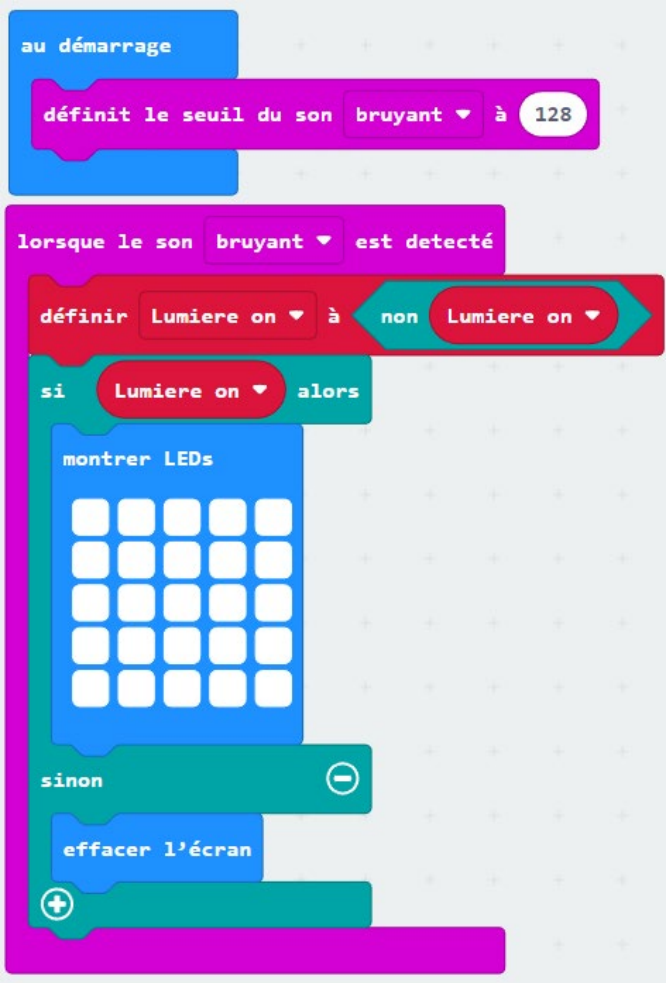
Il faut avant tout définir le niveau sonore de ce que l'on entend par « bruyant ». La valeur numérisée du capteur de son est comprise entre 0 et 255. Dans notre cas prenons 128.

Un nouveau concept est celui de booléen. Un booléen peut prendre 2 formes : soit Vrai (1), soit Faux (0).

Ici la variable est « Lumière On ». Si « Lumière On » est Vrai (1) alors on affiche les LEDs. Sinon on l'éteint.

À chaque fois que le son bruyant est détecté, on vient définir la variable « Lumière On » à l'inverse de l'état à laquelle elle se trouvait.

Le bloc permettant de changer l'état contraire/inverse de l'état initial est le bloc « non ».



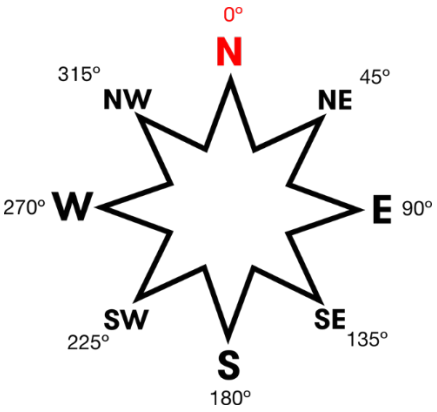
Une fois le programme téléversé, essayons la carte. Taper dans vos mains pour allumer l'écran de la carte, et retaper dans vos mains pour l'éteindre. Si ça marche, … on passe à la suite ! 😊

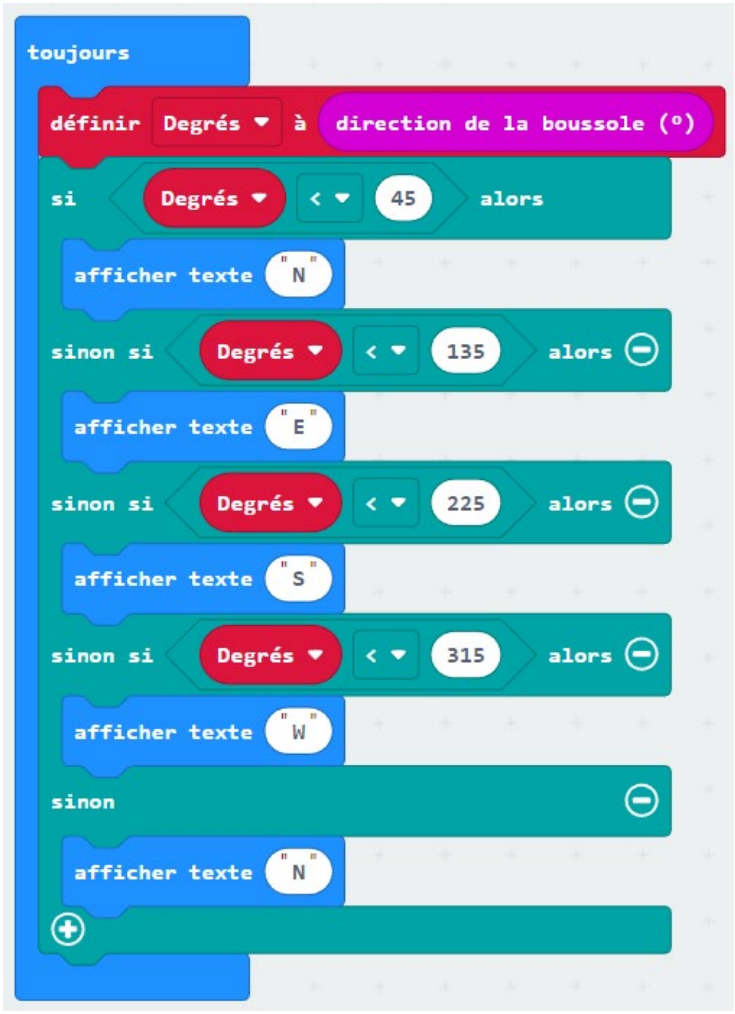
13. Boussole

Créer une variable Degrés, puis définir Degrés pour qu'il prenne la valeur de l'angle de la direction de la boussole.

Ensuite il suffit d'appliquer la condition Si avec un comparateur <.

Pour vous aider, et éviter le calcul des angles, je mets ci-dessous une rose des sables avec les degrés.





Une fois téléverser, votre carte peut afficher « FILL SCREEN ». Il faut la secouer jusqu'à ce que toutes les leds soient allumées.

Cela permet de calibrer le capteur de la carte.

Une fois calibré c'est maintenant une boussole captant le champs magnétique terrestre, Arriverez-vous à retrouver le nord ?

14. Envoyer une donnée via Bluetooth

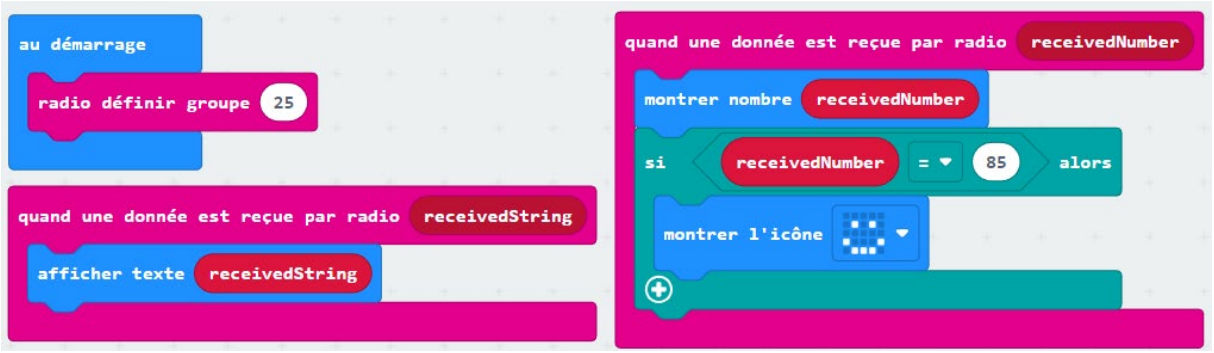
Apprenons à communiquer entre 2 cartes Microbit via la « Dent Bleu »… heu le Bluetooth. Il existe 2 types de données que nous pouvons envoyer via Bluetooth. Soit envoyer des nombres, soit envoyer une chaine de caractère/texte (string).

Comme pour un talkiewalkie, au démarrage, nous devons obligatoirement mettre les 2 cartes qui vont communiquer entre elles sur le même canal/groupe de communication. Nous avons le choix entre 0 et 255, soit 256 canaux de communication, ici 25.

Pour l'émetteur : Quand je presse « A » j'envoie le chiffre 85, « B » j'envoie Hello.

Pour le récepteur :

Quand la carte reçoit une chaine de caractère (string), elle l'affiche. Quand elle reçoit un nombre, le programme pourrait directement l'afficher. Ici on a choisi de comparer ce nombre. Si le nombre reçu est 85, alors on affiche un smiley.



Testez votre carte. Le Bluetooth passe-t-il à travers les murs ?

15. Recherche avalanche

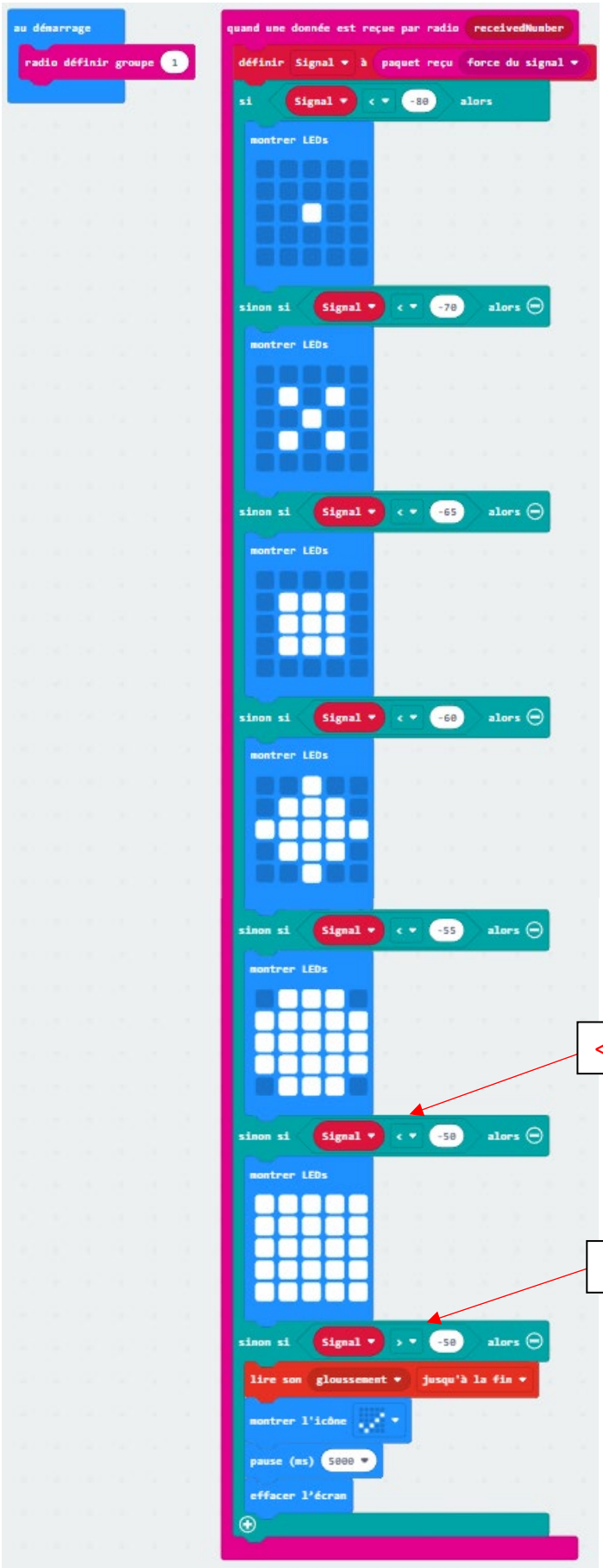
Je suis partie en montagne avec un groupe. Nous avons pris nos Microbit avec nous et nous avons synchronisé nos cartes sur le groupe/canal 1 avant de partir.

Brusquement, je suis pris sous une avalanche. Ma carte envoie le chiffre 0 pour toujours. (Je vais cacher ma carte dans le Fablab) avec une puissance d'émission à 5.

Voici ce qu'envoie ma carte (émettrice):

Les participants doivent faire un programme pour me retrouver sous l'avalanche grâce à la puissance du signal.

Le signal émet une puissance en décibel (dB) comprise entre -110dB et -42dB. Le nombre de décibel de la puissance du signal reçu doit être stocké dans une variable que nous appelons Signal.



Si le Signal est < -80dB on affiche un point sur l'écran.

Puis remplir l'écran au fur et à mesure avec les valeurs de -70dB, -65dB, -60dB, -55dB, -50dB.

Plus on se rapproche de la carte émettrice plus l'écran se rempli.

Voici ce que reçoit la carte (réceptrice).

Ouvrez une autre page avec MakeCode pour avoir les deux programmes en même temps.

Quand le signal est > à -50dB, alors nous avons trouvé la carte émettrice !! Nous pouvons déclencher un son, puis montrer une icone

En fonction de la sensibilité des cartes de réceptions, les valeurs de comparaison en décibels peuvent être modifiées (dans les limites de -110dB à -42dB) pour une plus grande précision.

16. Data recorder

Créons une variable Logging (Enregistrement) pour déterminer quand nous enregistrons nos informations ou non. Au démarrage de la carte, nous ne commençons pas notre enregistrement, notre variable est donc à Faux. Nous allons l'utiliser comme un booléen (Vrai ou Faux).

Je veux éteindre et allumer mon enregistrement en appuyant sur le bouton A. L'on procède comme avec la « Lampe clap » et le bloc « non » qui vient inverser/ mettre à l'état contraire ma variable.

On joue un son au démarrage de la carte tout en laissant afficher une icône pour montrer que la carte est active.

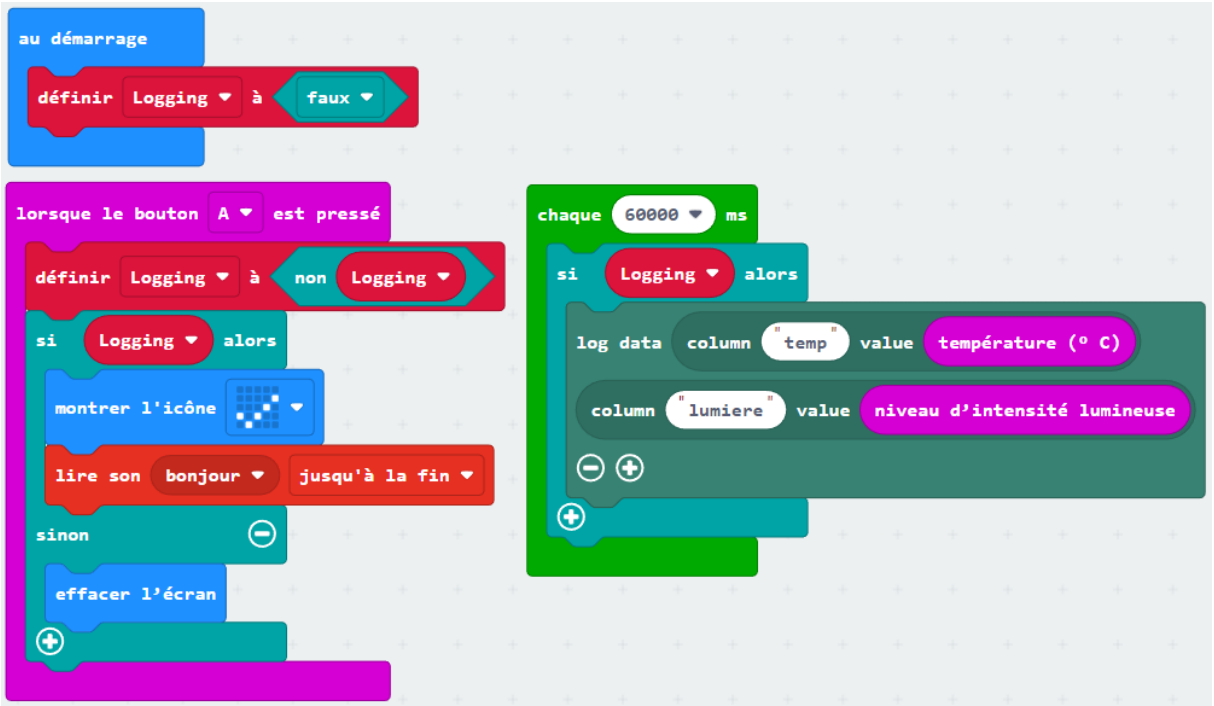
Puis toutes les minutes je viens enregistrer les valeurs de la température (en degré) et l'intensité lumineuse (de 0 à 255). Pour cela nous devons installer une extension. **Il faut donc allez dans « Extension » et cliquer sur « Datalogger ».**

datalogger

Data logging to flash memory.
micro:bit (V2) only.

Nous pourrions aussi enregistrer ici l'intensité de bruit, ou encore l'inclinaison de la carte...

Attention, ne pas mettre d'accent dans les variables !!!



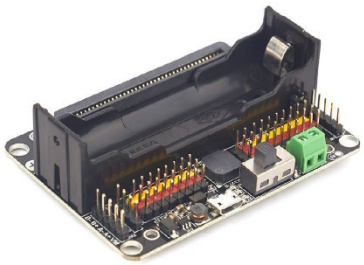
Comment récupérer les informations enregistrer ?! Il suffit de se reconnecter à la Microbit via un PC et d'aller chercher le fichier My_Data. 😊

Les chiffres du fichier My_Data contiennent la température en C° et le niveau d'intensité lumineuse (compris entre 0 et 255. 0 pour la pénombre totale, et 255 en pleine lumière avec le capteur saturé).

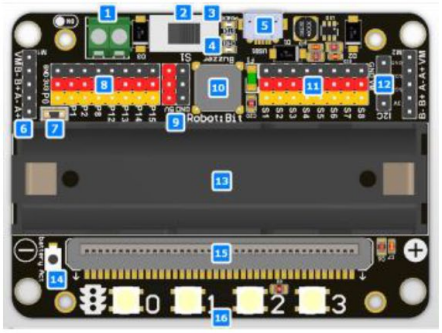
On peut ensuite mettre les données dans Excel pour tracer des courbes et voir l'évolution dans le temps.

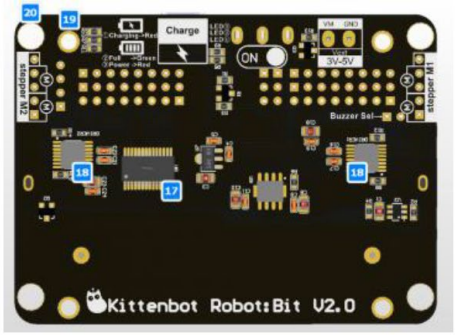
17. Introduction à la RobotBit

Il est temps d'utiliser une carte « shield » (bouclier). C'est une carte qui va venir se greffer à notre Micro:Bit afin de lui donner des fonctionnalités supplémentaires.



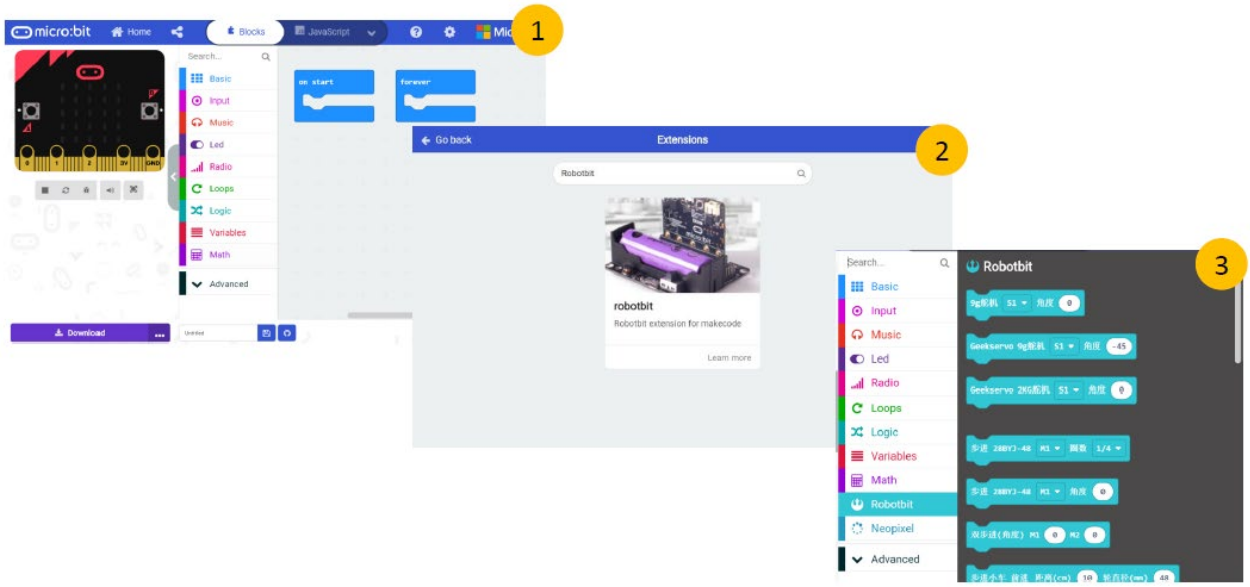
Grace à cette nouvelle carte nous pouvons brancher des servomoteurs, des moteurs CC, et des petits moteurs pas à pas. Nous pouvons aussi piloter 4 LED RGB (Red/Green/Blue).

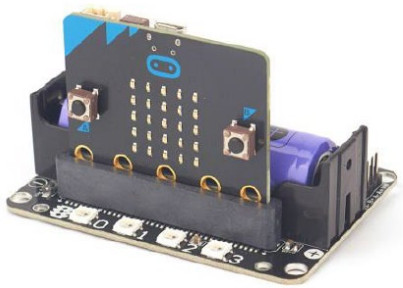




1. External power terminal (5V=3A)	2. power switch	3. Power indicator	4. battery indicator	5. Charging interface (Micro USB)
6. 4 Motor Interface / 2 stepper motor Interface	7. Buzzer sw	8. 8 I/O pins	9. VCC & GND	10. Passive Buzzer
11. 8 channel servo interface	12. I2C	13. 18650 Lithium battery holder	14. Power activation button	15. 40P edge connector
16. 4 rgb LED (ws2812)	17. PWM IC	18. motor driver IC	19. M3 copper column fixing hole	20. LEGO standard hole

Dans MakeCode nous devons ajouter les nouvelles fonctionnalités. Pour ce faire, dans les bibliothèques allez dans « Extension » puis chercher « RobotBit ». De nouvelles options apparaissent.



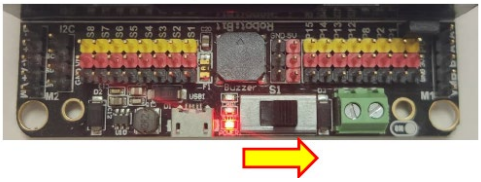


Il est important de mettre la carte Micro:Bit dans le bon sens. Le même sens que sur la photo. Les LEDs de la carte Micro:Bit du même côté que les LEDs de la carte RobotBit.

Une fois la batterie 18650 installée, nous pouvons allumer la carte. Si elle ne s'allume pas, vous appuyez sur le bouton derrière la batterie.

La carte peut aussi être alimenté par le port USB.

■ Turn on the power



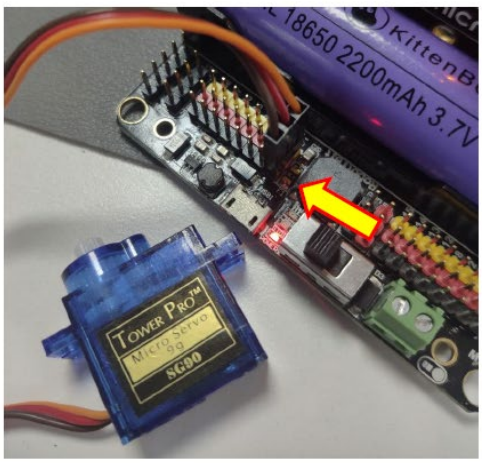
■ If you turn on the power switch and the power does not light you need to click the power activation button at this time

18. Utilisation d'un servomoteur

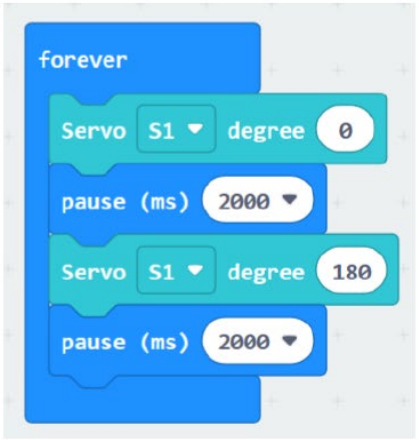
Un servomoteur utilise des angles compris entre 0 et 180° pour se mouvoir. Afin de préserver la mécanique de notre servomoteur, nous allons limiter les angles entre 10° et 170°.

Ici je propose que nous réalisation un mouvement de va et vient comme pour faire un balai d'essuie-glace de voiture avec une pause de 2s entre chaque mouvement.

■ Connect servo such SG90 to S1, please pay attention to the line order



■ Control the angle of 9g servo



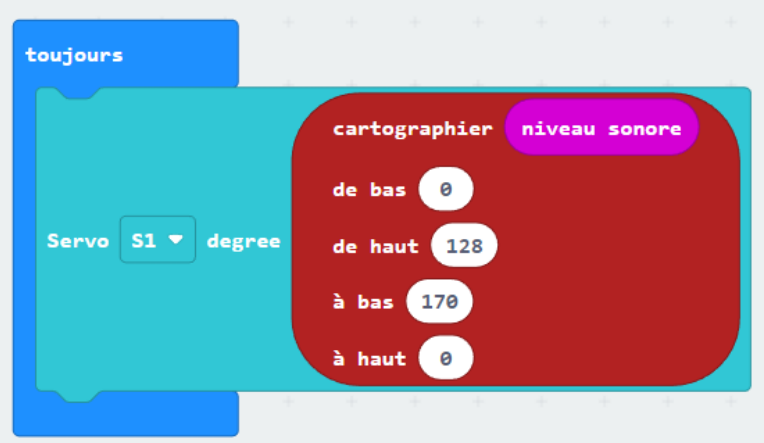
19. Sonomètre avec servo – Mapping d'une entrée

Ici nous allons utiliser le bruit ambiant pour faire bouger proportionnellement notre servomoteur au bruit ambiant. Pour rappel, notre servomoteur va de 10° à 170°. Plus il y aura du bruit, plus notre servomoteur va augmenter son angle. Si peu de bruit, il va alors rester à 10°.

Cette action s'appelle le mapping. Nous allons transformer une valeur d'entré (analogique ou numérique) en une valeur de sortie en créant un rapport de proportion automatique entre les deux. Le bloc « mapping/cartographier » se trouve dans la bibliothèque « Broches ».

La Micro:Bit capte un son qui va être converti de l'analogique vers le numérique. La valeur du « niveau sonore » va être compris entre 0 et 255.

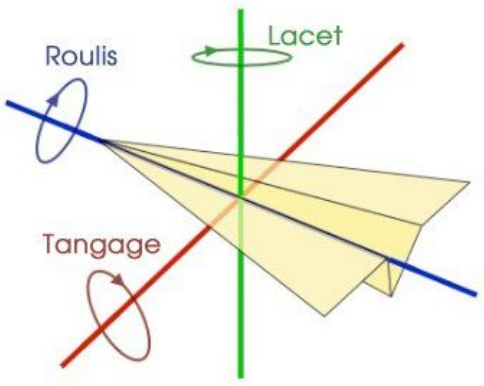
Partons du principe que la valeur de 128 sera considérée comme forte pour nous. Nous voulons que lorsque le micro enregistre un niveau sonore de 0, l'angle de notre servomoteur soit de 10°. Puis si le microphone capte un niveau sonore de 128 ou plus, l'angle soit à 170°.



Pour inverser la direction de notre servomoteur, on peut inverser les angles.

20. Servomoteur/Inclinaison carte + Lien série

Notre carte a un capteur d'inclinaison qui va nous permettre de nous dire quand la carte est penchée et sur quel axe elle est penchée.

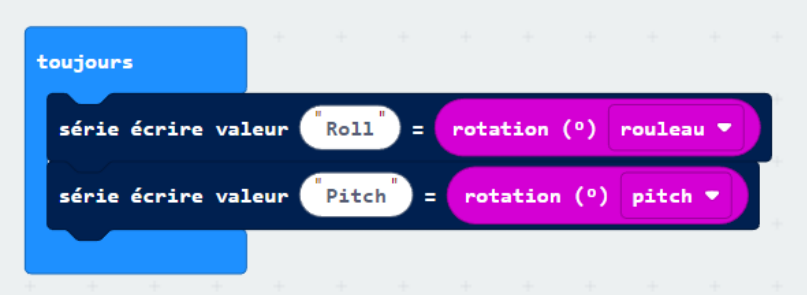


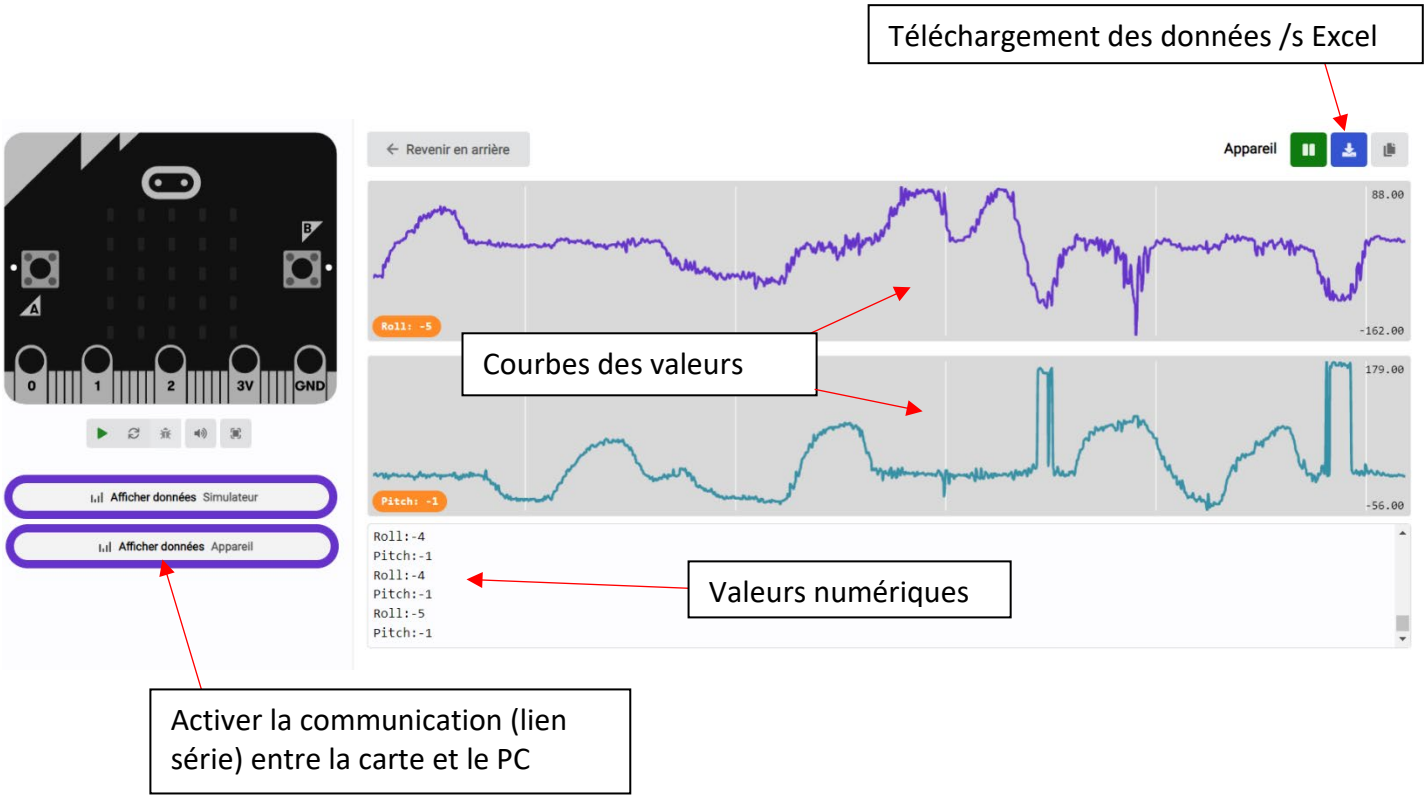
La carte va pouvoir détecter le Roulis (Roll) et le Tangage (Pitch).

Il est important de connecter la carte et qu'elle soit visible comme connectée /s Make Code.

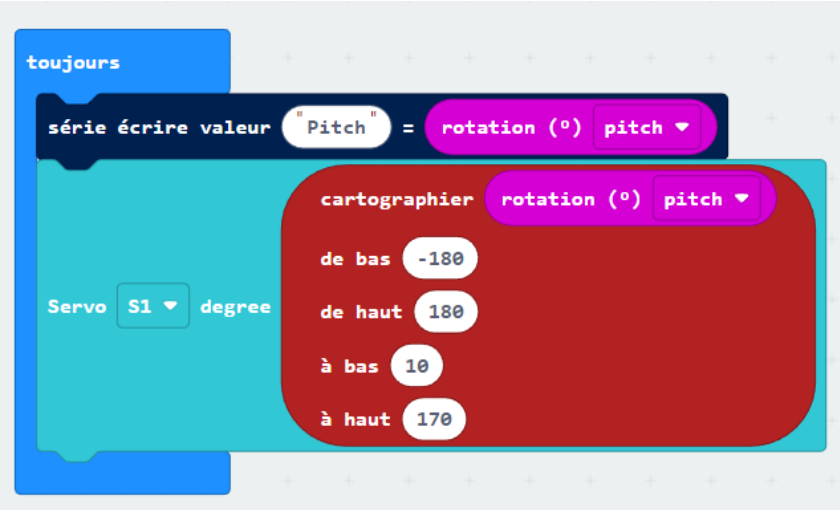
Attention cependant, le navigateur web est susceptible de bloquer le lien série en ne donnant pas accès à l'USB. Si cela ne marche pas tester avec un autre navigateur.

Téléverser le programme dans la carte. Une fois fait, afficher les données de l'appareil.





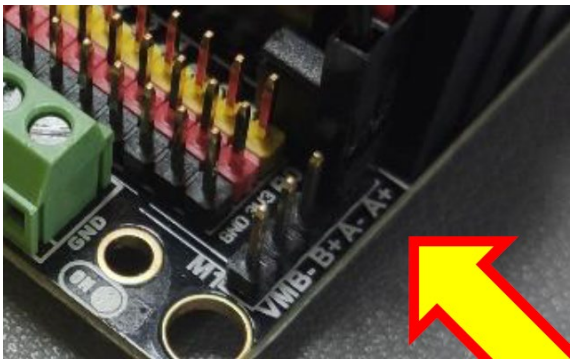
Maintenant que nous savons utiliser le liens série et voyons que les valeurs changent en fonction de l'inclinaison de la carte… utilisons les valeurs du pitch pour faire bouger un servomoteur !

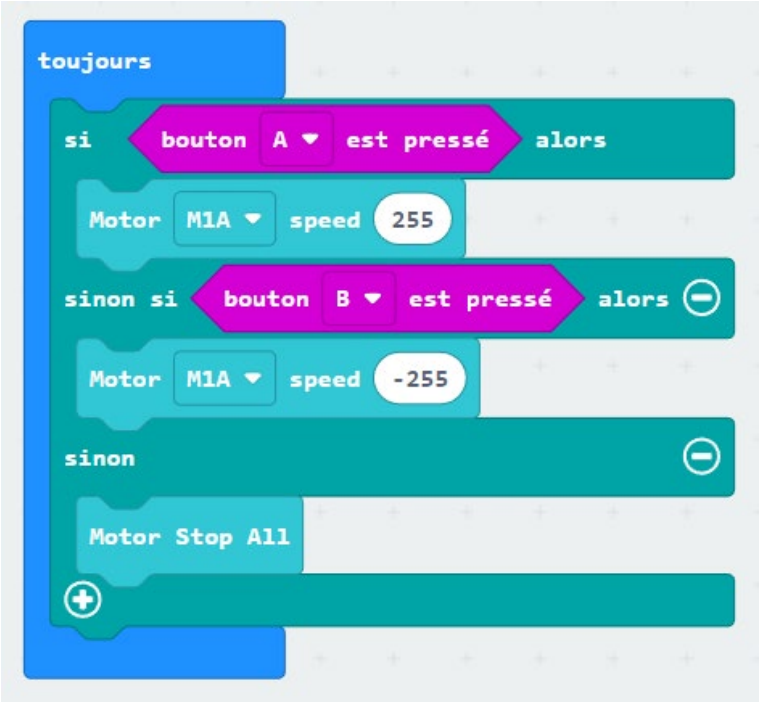


La valeur du Pitch varie de -180° à 180°, et pour rappel le servomoteur va de 10° à 170°. Nous allons donc faire un mapping en conséquence. On peut aussi laisser la carte brancher au PC pour quelle puisse continuer à envoyer ses valeurs et ainsi voir les données et le graphique sur l'écran du PC.

21. Utilisation moteur

À présent, utilisons un moteur à courant continu (DC). Il y a 4 emplacements moteurs sur la RobotBit (M1A, M1B, M2A, M2B) chacun des emplacements étant composés d'une borne + et une broche - . Sur la photo le moteur est branché sur M1A.

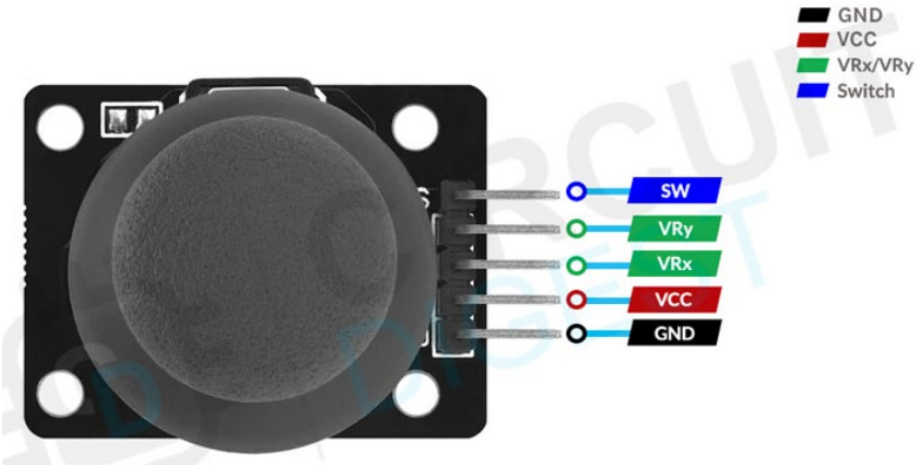




Avec ce programme, si nous appuyons sur le bouton A, il tourne dans un sens. Si nous appuyons sur le bouton B il tourne dans l'autre sens. Dans les autres cas il s'arrête.

22. Moteur avec mapping joystick

Imaginons que je souhaite faire varier le moteur de façon progressive avec l'utilisation d'un joystick. Regardons un peu à quoi correspondent les différentes broches du joystick.

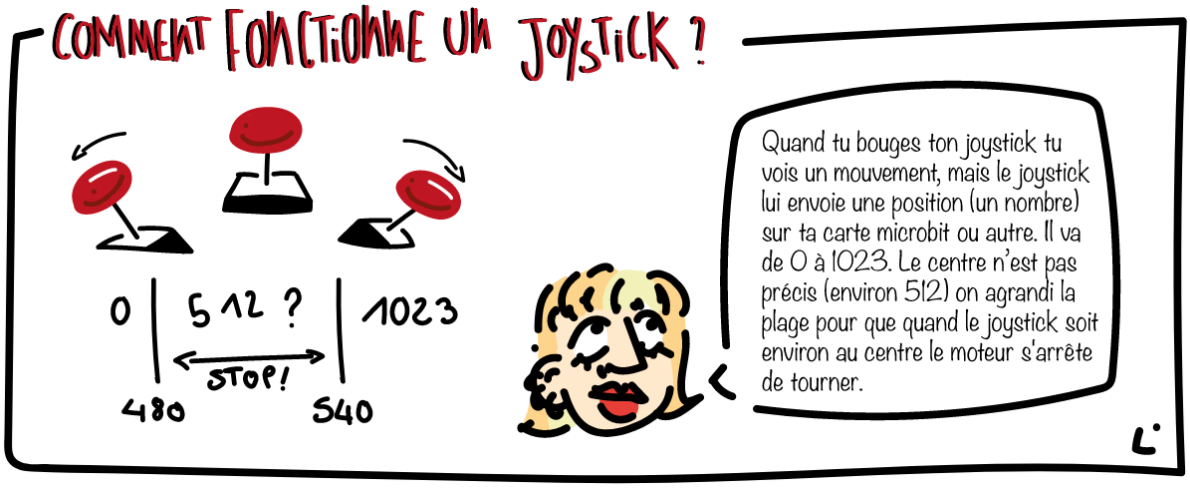


Le GND est le -, c'est la masse. Elle se branchera sur une broche noire de notre carte RobotBit.

Le VCC est l'alimentation. Ici nous allons alimenter notre joystick en +3.3V. **Ne pas le brancher sur le 5V, cela pourrait endommager la carte Microbit qui n'accepte que du 3.3V en entré.** Le 3.3V sont les broches rouges entre les broches noires et jaunes.

Le SW est un switch, c'est-à-dire un bouton que l'on peut actionner en appuyant sur le joystick. Nous n'allons pas nous en servir ici.

VRx et VRy sont les tensions variables que nous allons récupérer pour connaitre la position de notre joystick. Un joystick est en fait composé de 2 résistances variables qui va ici varier entre 0V et 3.3V. La carte va récupérer cette tension (analogique) et la transformer pour donner un signal compris entre 0 et 1023.



Utilisons qu'un axe, l'axe X pour simplifier notre montage et notre programme.

Les broches P0, P1 et P2 sont les broches analogiques, mais nous allons laisser P0 de côté car elle est utilisée pour le buzzer de la carte RobotBit.