microDEFI

Défis de programmation microPAD – microMOUV - microSCORE

D-MD-2025

Édité par la société A4 Technologie
5 avenue de l'Atlantique - 91940 Les Ulis
Tél. : 01 64 86 41 00 - www.a4.fr

Logiciels, programmes, manuels utilisateurs téléchargeables gratuitement sur www.a4.fr

microPAD

microSCORE

Option
barre 10 LED RVB

microMOUV

Option
afficheur 4 digits

Pack microSCORE,
microPAD, microMOUV

Livre micro:bit

SOMMAIRE

Système didactique microDEFI

Introduction

microDEFI est un système didactique basé sur le thème du jeu. Il est composé des 3 matériels communicants **microMOUV**, **microPAD** et **microSCORE** équipés chacun d'une carte micro:bit.
Ces trois matériels sont utilisés séparément ou en interaction les uns avec les autres pour imaginer et concevoir une infinité de jeux ou défis ludiques.

Leur programmation est simple et facile à mettre en œuvre avec les plateformes micro:bit **MakeCode** pour les programmer en **blocs** ou en **Python** et micro:bit **Create AI**, pour créer des blocs personnalisés de Machine Learning et les intégrer dans les programmes.

La **communication radio** est très simple à mettre en œuvre, elle offre de nombreuses possibilités d'interactions entre les différents matériels.

Ce dossier propose une série d'exemples progressifs pour créer des jeux. Les **programmes** proposés sont réalisés en **blocs avec MakeCode** et les **modèles d'entraînement IA** sont réalisés avec **micro:bit Create AI.**
L'intention de ces exemples est de maîtriser les aspects techniques liés à la programmation des différents matériels, de proposer des idées de jeux et de susciter une démarche d'ingénierie simple autour d'un projet commun.

L'ensemble des programmes présentés est disponible sur www.a4.fr dans la rubrique « Ressources » microDEFI

Les élèves pourront s'organiser en groupes pour imaginer et concevoir un jeu utilisant :

- **microPAD (interface de commande / affichage)**
- **microMOUV (détection de mouvements / gestes)**
- **microSCORE (gestion du score / temps / manches)**

Communication radio

Gestion d'un projet microDEFI

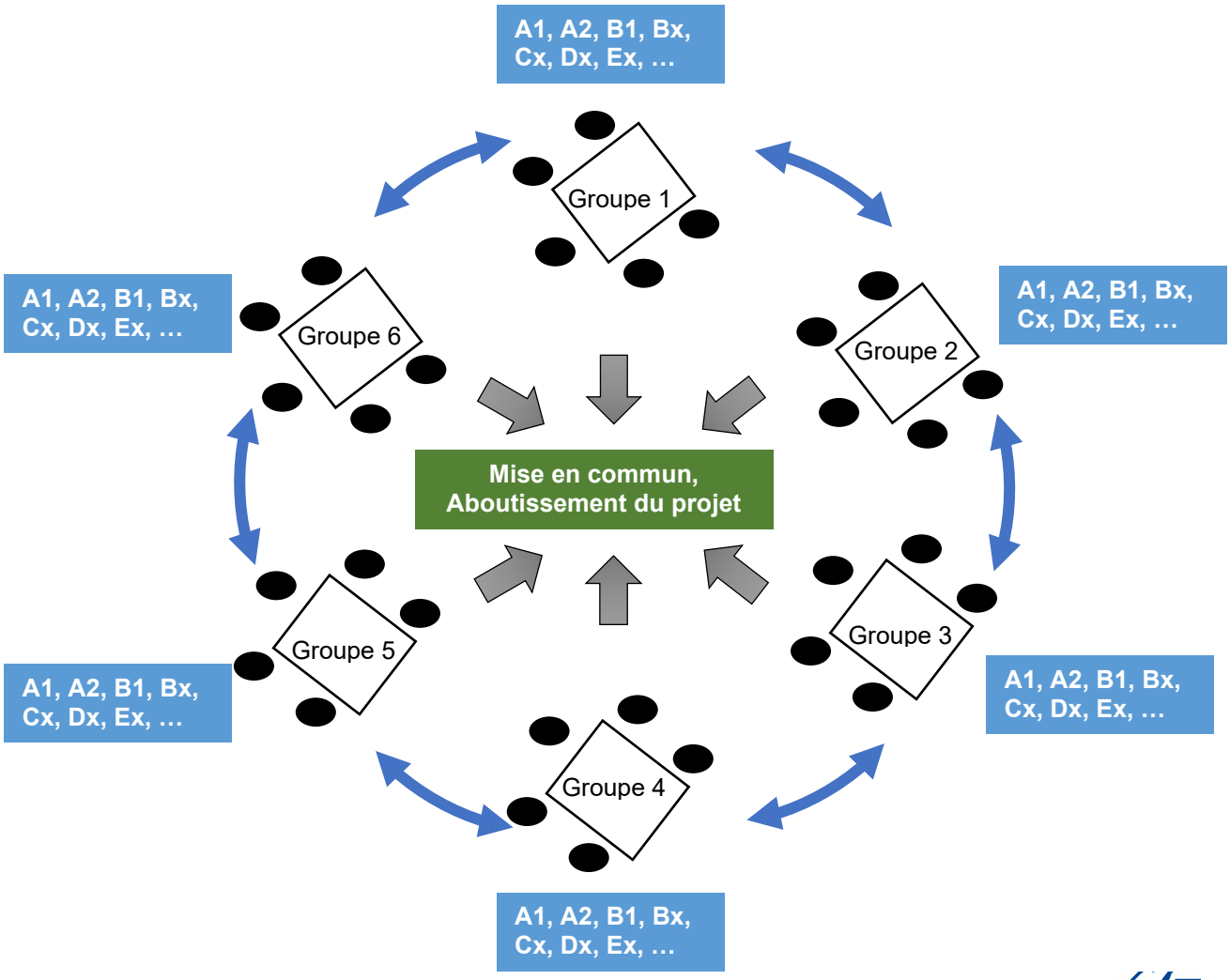
La programmation n'est qu'une phase du projet. Pour mener le projet de manière efficace et aboutir à un système fonctionnel et fiable, il convient d'en cerner tous les aspects et de s'organiser avant de se lancer dans la programmation.
Le tableau ci-dessous suggère un mode d'organisation et une liste de tâches pour créer un jeu. Les tâches peuvent être regroupées ou simplifiées selon la complexité du jeu ou le niveau de la classe.

EXEMPLE :

Phase	N°	Tâche	Objectif	Livrable / trace écrite
A - Lancement	A1	Gestion du projet	Répartir les rôles dans le groupe (chef de projet, responsable programmation, responsable tests, responsable communication), construire un mini planning.	Fiche "équipe projet" avec la liste des membres, leurs rôles et un planning simplifié.
A - Lancement	A2	Découverte du matériel (microPAD, microMOUV, microSCORE)	Comprendre rapidement ce que font chaque module et ses capteurs/actionneurs.	Fiche d'inventaire du matériel + schéma simple d'assemblage.
B - Imaginer	B1	Brainstorming collectif (dans chaque groupe, puis mise en commun classe)	Trouver des idées de jeux possibles, à partir des contraintes de temps, de matériel et de niveau.	Carte mentale ou liste d'idées de jeux, avec 1 ou 2 idées retenues.
B - Imaginer	B2	Scénarisation du jeu	Définir l'histoire / contexte du jeu, le but, le nombre de joueurs, la durée d'une partie, les conditions de victoire / défaite.	Fiche "scénario de jeu" (début, déroulement, fin, conditions de victoire).
B - Concevoir	B3	Cahier des charges	Traduire l'idée de jeu en contraintes techniques : quels capteurs utiliser, quels affichages, quels sons, quelles limites de temps, quelles règles d'IA.	Cahier des charges d'1 à 2 pages : objectifs, contraintes, matériel utilisé, critères de réussite.
B - Concevoir	B4	Définition des données du programme	Lister les données utiles au fonctionnement du jeu : variables (score, temps, niveau), capteurs (accélération, mouvements), seuils, états du jeu.	Tableau des variables (nom, rôle, type, valeurs possibles) + liste des capteurs utilisés sur microPAD / microMOUV / microSCORE.
B - Concevoir	B5	Algorigramme (organigramme du programme)	Décrire le fonctionnement du jeu sous forme de blocs : démarrage, boucle de jeu, prise en compte des gestes, calcul du score, fin de partie.	Algorigramme dessiné (ou pseudo-code) pour : 1) démarrage du jeu, 2) boucle principale, 3) fin de partie.
B - Concevoir	B6	Modèles d'entraînement IA (si IA utilisée)	Choisir ce que l'IA doit reconnaître (gestes, mouvements, situations), collecter des exemples, organiser les données d'entraînement et de test.	Fiche "modèle IA" avec : classes à reconnaître, nombre d'exemples par classe, protocole d'enregistrement et critères de réussite.
C - Programmer	C1	Programmation	Programmer le jeu en blocs (MakeCode micro:bit), en respectant l'algorigramme et le cahier des charges. Organiser le code en modules : gestion capteurs, IA, score, affichage.	Programme(s) MakeCode commentés, sauvegardés et partagés au sein du groupe.
C - Programmer	C2	Tests unitaires des programmes	Tester séparément chaque partie du programme (lecture des capteurs, calcul du score, affichage, IA) avec des cas simples et répétables.	Fiche de tests unitaires : pour chaque module, une liste de cas de test + validation "OK / à corriger".

C - Programmer	C3	Mise en commun des programmes (intégration)	Assembler les différentes parties du code (capteurs, score, IA, affichage) dans un programme final cohérent pour le jeu.	Version intégrée du programme du jeu, avec une note “version v1, v2…” pour suivre les corrections.
D - Valider	D1	Cahier de recette fonctionnelle	Définir les scénarios de tests complets du jeu : déroulement normal, erreurs possibles, comportements attendus de microPAD / microMOUV / microSCORE.	Cahier de recette : tableau “scénario de test / action / résultat attendu / résultat obtenu / OK ou NOK”.
D - Valider	D2	Recette fonctionnelle (campagne de tests)	Faire jouer d’autres élèves / groupes au jeu en suivant le cahier de recette, mesurer le respect du cahier des charges et la jouabilité.	Cahier de recette rempli, avec les anomalies repérées et les améliorations proposées.
E - Finaliser	E1	Améliorations et corrections	Corriger les bugs majeurs, ajuster la difficulté du jeu, améliorer lisibilité des scores ou indications visuelles/sonores.	Nouvelle version du programme et note de synthèse des améliorations réalisées.
E - Finaliser	E2	Communication & présentation du jeu	Préparer une courte présentation : nom du jeu, principe, règles, rôle de microPAD/microMOUV/microSCORE, rôle de l’IA, difficultés rencontrées.	Affiche, fiche de présentation ou diaporama, plus éventuellement une petite notice d’utilisation du jeu.

Exemple d’organisation des groupes et de répartition des tâches

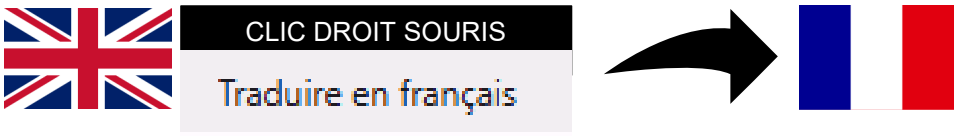


Prérequis

- Disposer des environnements de programmation MakeCode et/ou micro:bit Create AI
- Maîtriser le fonctionnement de base des logiciels de programmation et de la carte micro:bit

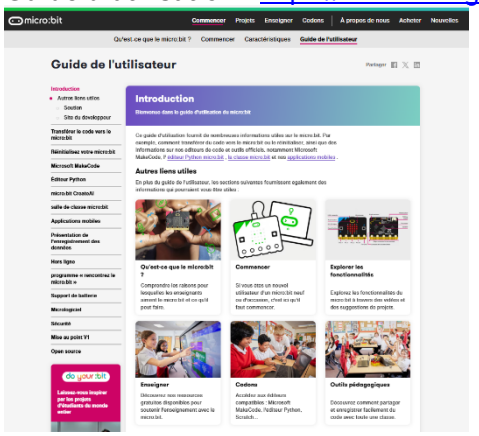
Documentations carte micro:bit

ASTUCE : version française des documents en anglais

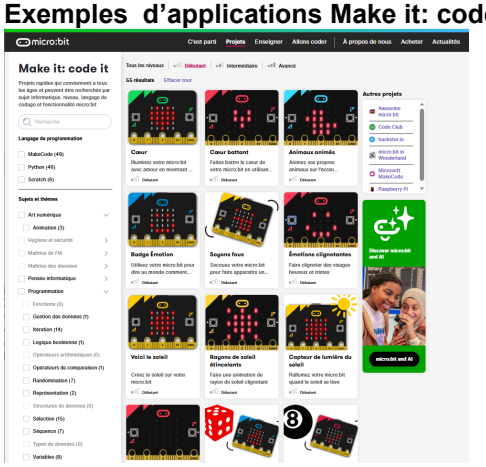


Site du fabricant : https://microbit.org/get-started/what-is-the-microbit/

Guide d'utilisation : https://microbit.org/get-started/user-guide/introduction/



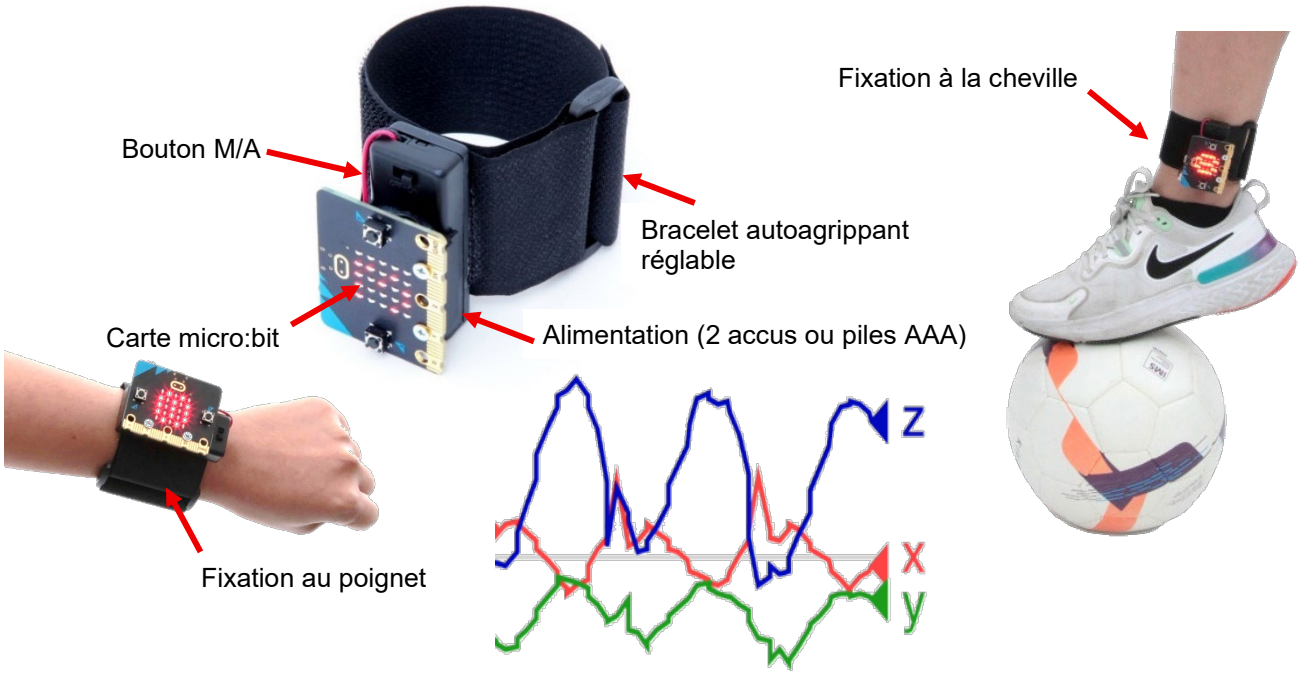
Exemples d'applications Make it: code it : https://microbit.org/fr/projects/make-it-code-it/



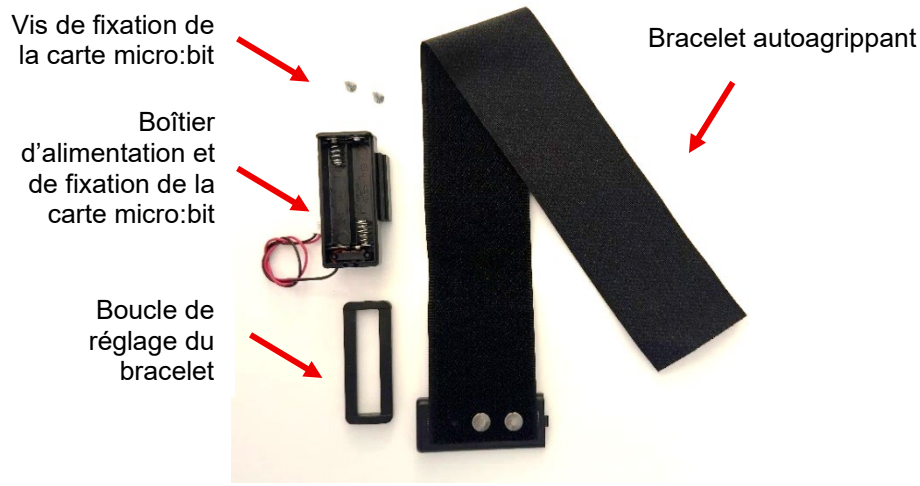
Mise en service du bracelet connecté microMOUV

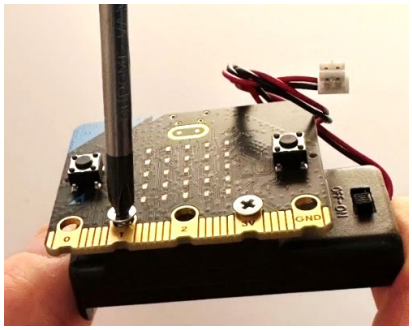
microMOUV est un bracelet connecté qui détecte et différencie de manière autonome des mouvements appris avec l'environnement d'**Intelligence Artificielle « micro:bit CreateAI* »**. Il est équipé d'une carte micro:bit alimentée par deux piles ou accus AAA. Son bracelet large peut être ajusté pour le porter au poignet ou à la cheville. Sa carte micro:bit est programmée pour détecter des mouvements comme (course, sauts, gymnastique, etc.) pour réaliser des défis sportifs. Il peut comptabiliser un nombre de mouvements détectés, émettre des sons, afficher un score et communiquer avec le panneau d'affichage de microSCORE ou le pad de jeu microPAD.

* CreateAI génère à la demande, par apprentissage, des blocs de programmation dédiés à la détection de mouvements

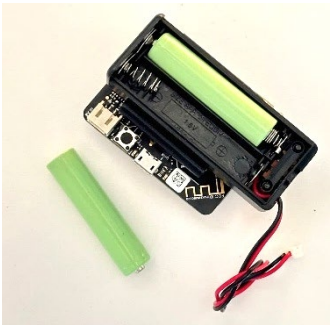


Montage du kit microMOUV





À l’aide des 2 vis 3x6,5 mm, fixer une carte micro:bit V2 sur son support

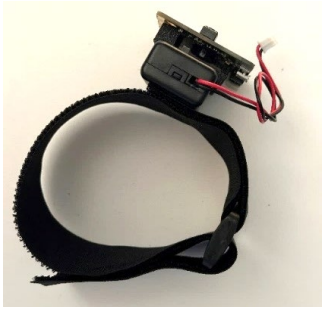


Insérer 2 piles ou accus AAA dans le boîtier d’alimentation

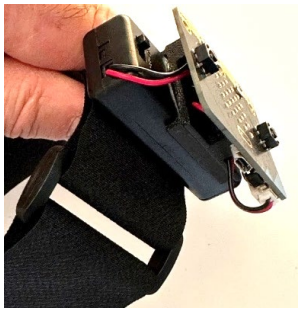
Insérer la boucle dans le bracelet

Rabattre le bracelet sur lui même

Insérer le couvercle du boîtier d’alimentation



Passer l’extrémité du bracelet dans la boucle pour ajuster son serrage



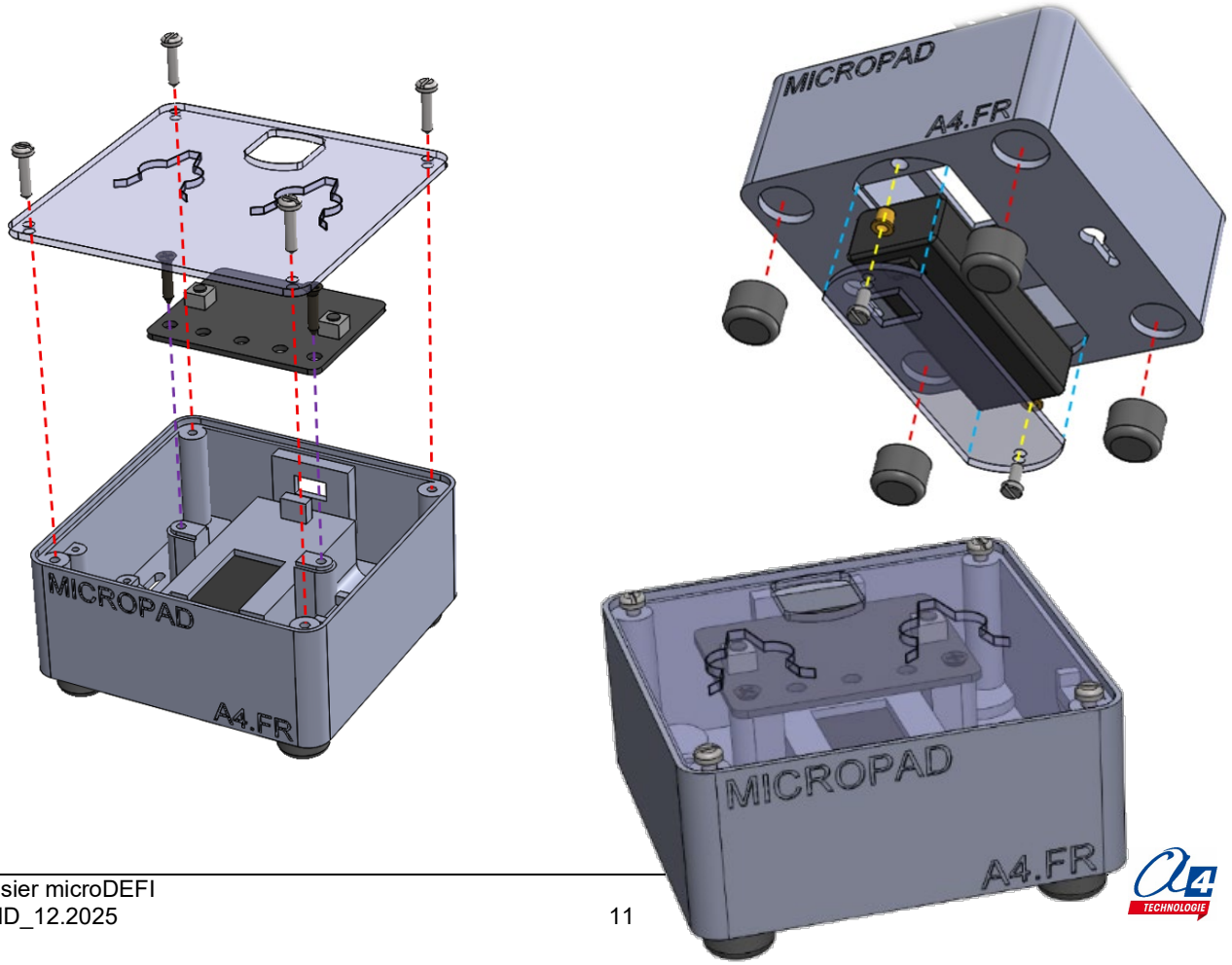
Insérer le connecteur d’alimentation de la carte micro:bit

Mise en service du Pad de jeu microPAD

microPAD est un Pad de jeu qui réagit aux chocs ou aux déplacements appris avec l'environnement d'**Intelligence Artificielle « micro:bit CreateAI* »**. Il est **programmable** avec une carte **micro:bit** et dispose d'une **alimentation autonome.** Il s'utilise seul ou en interaction avec d'autres Pads posés sur une table ou fixés au mur. Il est personnalisable en y intégrant des visuels imprimés visibles au travers de sa vitre de protection transparente. Il est prévu pour accueillir des options comme un afficheur 4 digits ou une barre de 10 LED RVB. Sa carte micro:bit est programmée pour réaliser une infinité de jeux (Quizz, Arbitre de jeu, Réflexe, Chronomètre, Morpion, Applaudimètre, …). Il peut émettre des sons, afficher un score et communiquer avec le panneau d'affichage de microSCORE ou le bracelet connecté microMOUV.

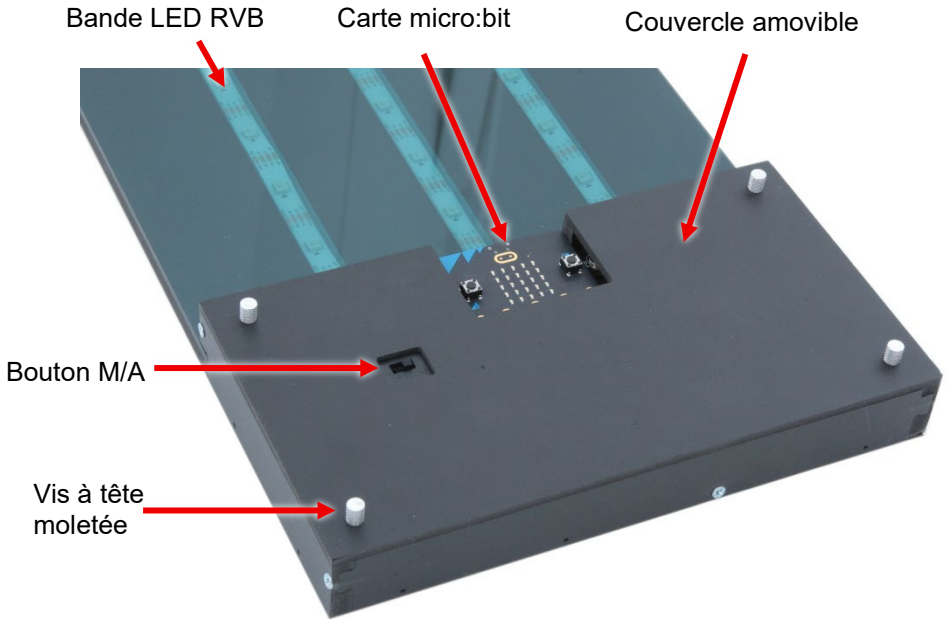
Montage du kit microPAD

Fixer la carte micro:bit à l'aide des 2 vis à tôle tête fraisée 3 x 16
Insérer 2 piles ou accus AAA dans le boîtier d'alimentation
Intégrer le boîtier d'alimentation, le connecter à la carte micro:bit
Intégrer éventuellement un visuel imprimé avant de refermer la vitre de protection
Positionner la vitre pour la fixer à l'aide des 4 vis à tôle tête cylindrique 3 x 13
Positionner la trappe de piles pour la fixer à l'aide de 2 vis pas métrique M3 x 6
Coller les pieds amortisseurs antidérapants aux 4 coins du boîtier micro:PAD.

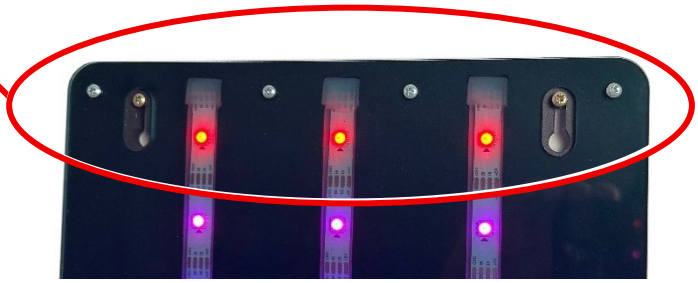


Mise en service du panneau d'affichage microSCORE

microSCORE est un panneau d'affichage équipé avec **3 bandes de 30 LED RVB**. Il est équipé d'une carte **micro:bit** et dispose d'une **alimentation autonome** par trois piles ou accus AA. Il s'utilise seul pour compter des points, afficher un décompte de temps, visualiser un niveau sonore …
Il peut émettre des sons, communiquer avec microPAD ou microMOUV pour afficher des scores.
Il se pose au sol, sur une table ou est accroché au mur.

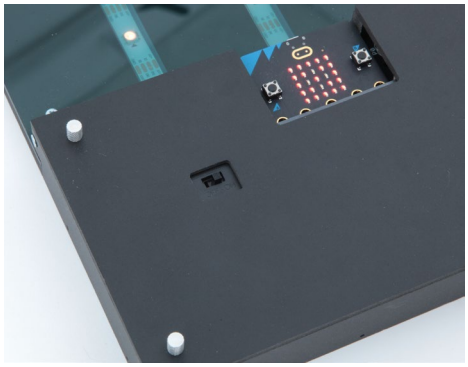


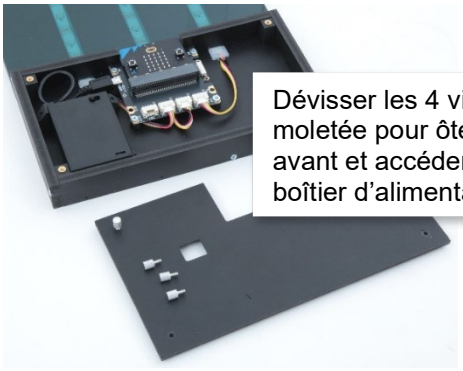
Trous de fixation pour suspendre microSCORE au



Alimentation intégrée avec 3 accus ou piles AA
ou
Alimentation externe par câble micro-USB et chargeur USB ou Power Bank externe

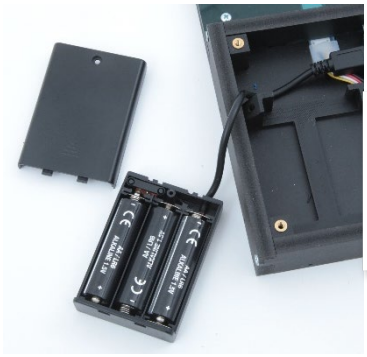
Mise en service



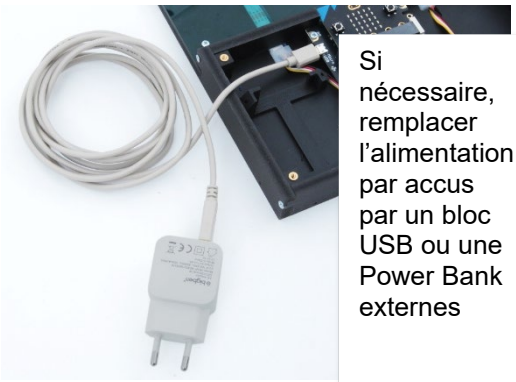


Dévisser les 4 vis à tête moletée pour ôter la face avant et accéder au boîtier d'alimentation

Oter le boîtier d'alimentation de son logement, exercer une pression légère sur son couvercle et le faire glisser dans le sens de la flèche pour l'ouvrir

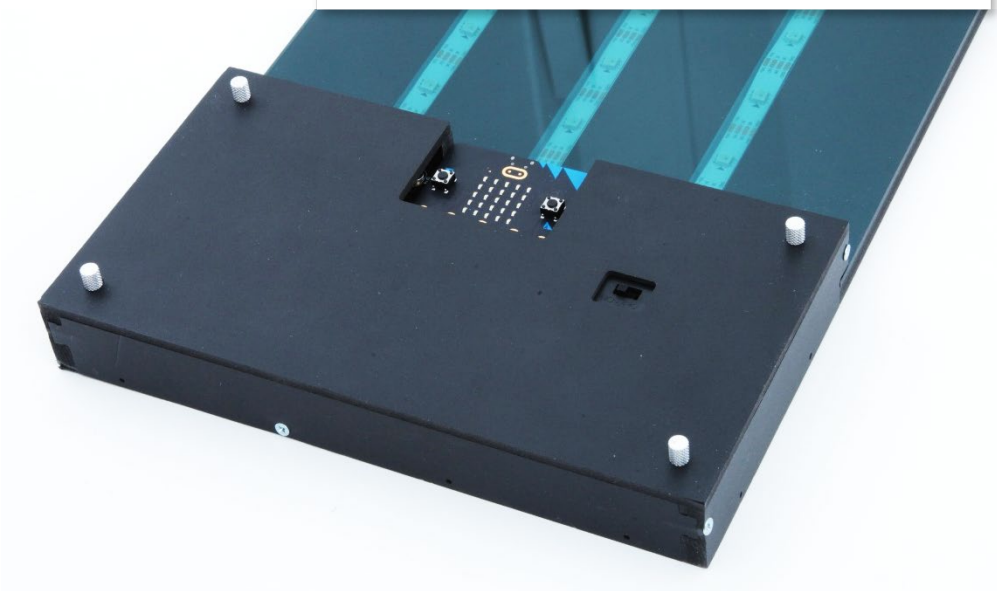


Insérer 3 accus ou piles AA, refermer le couvercle, repositionner le boîtier d'alimentation dans son logement



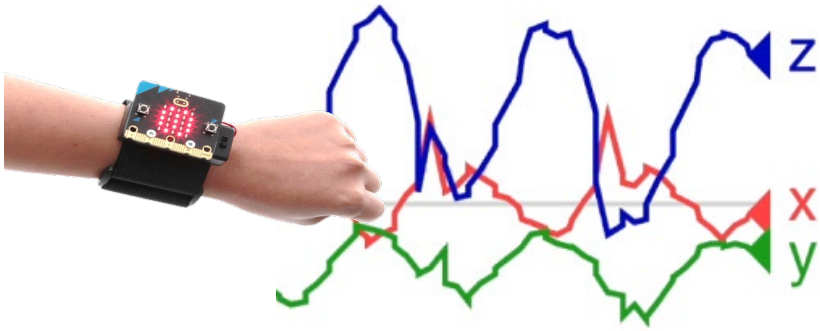
Si nécessaire, remplacer l'alimentation par accus par un bloc USB ou une Power Bank externes

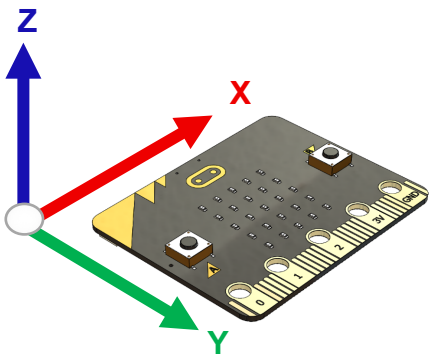
Clipser la face avant dans son logement, remettre en place les 4 vis à tête moletée

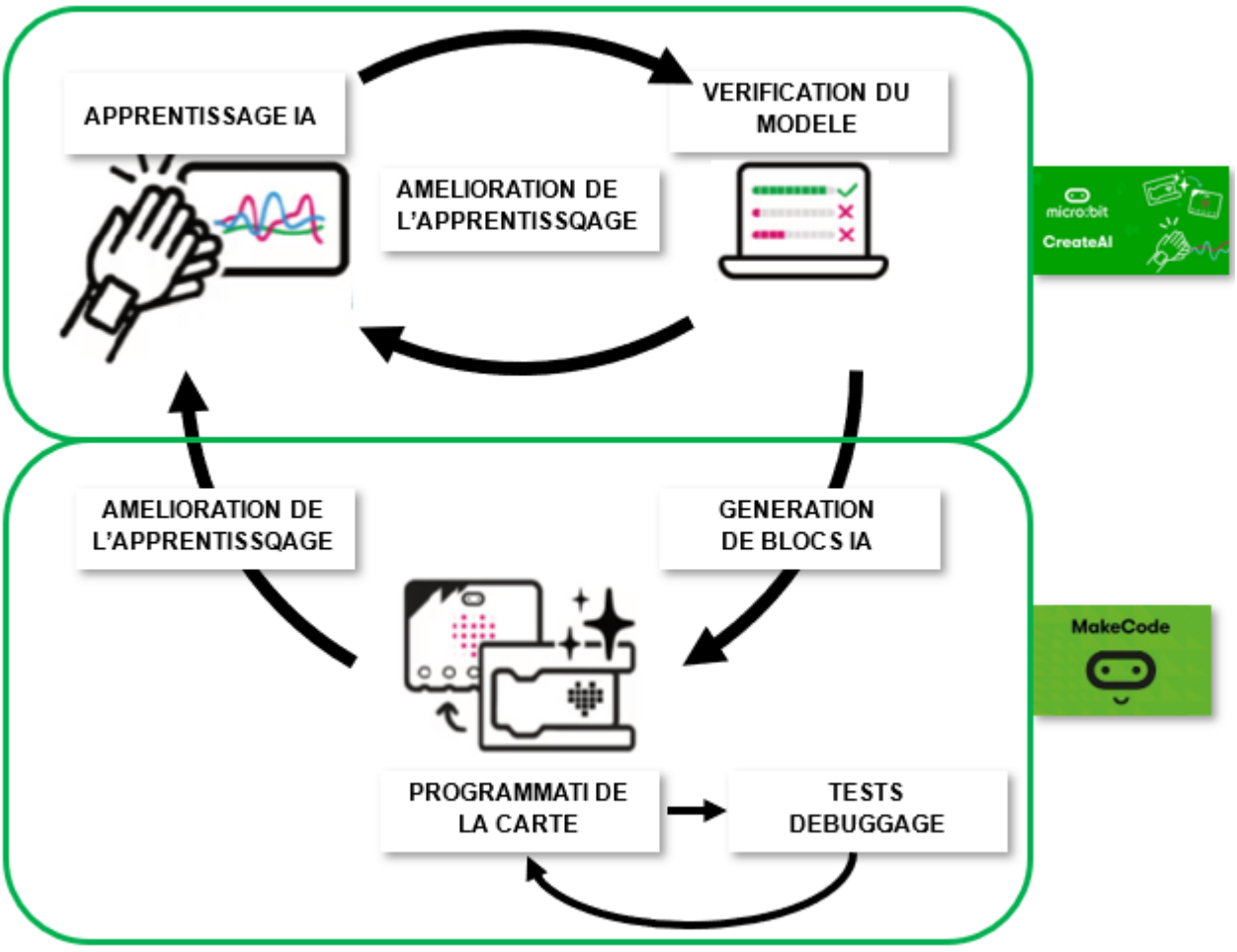


Prise en mains de micro:bit CreateAI

Micro:bit CreateAI est un environnement d'apprentissage par Intelligence Artificielle qui s'appuie sur les données d'accélération de la carte micro:bit selon les 3 axes X, Y, Z. L'IA est alimentée par des séries de mouvements et génère des blocs de programmation facilement exploitable dans Makecode pour reconnaître les séquences souhaitées.
Cet environnement est parfaitement adapté à l'utilisation du bracelet connecté microMOUV porté au poignet ou à la cheville en vue de détecter et de différencier différents mouvements .







Plateforme internet micro:bit CreateAI : https://createai.microbit.org/

Guide d'utilisation micro:bit CreateAI : https://microbit.org/get-started/user-guide/microbit-createai/

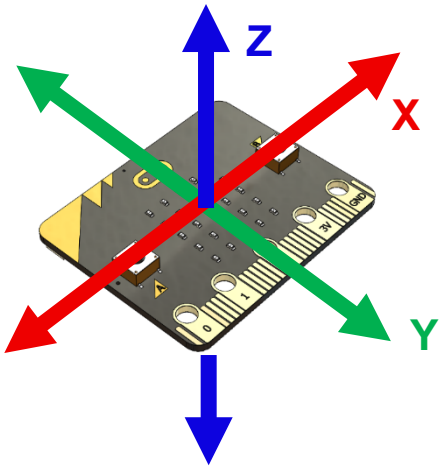
Principe de fonctionnement

micro:bit CreateAI est un environnement qui permet de générer par **apprentissage supervisé** des blocs de programmations destinés à détecter et à différencier des mouvements de la carte micro:bit. En maintenant la carte au poignet ou à la cheville, on détecte par exemple le mouvement de balancier du bras pendant une course, le mouvement d'une jambe qui tape dans un ballon, etc.
Les données des accélérations produites selon les 3 axes X, Y, Z par le capteur embarqué dans la carte micro:bit sont exploitées par l'algorithme d'IA déjà préentraîné pour détecter différents mouvements.
L'utilisateur enregistre à plusieurs reprises le mouvement qu'il souhaite détecter ; l'IA analyse les différentes séquences d'accélérations et effectue des traitements mathématiques (moyenne, variation, durée, etc.) pour les comparer et déterminer les corrélations qui existent entre des séquences similaires.
Lors de l'apprentissage, l'utilisateur visualise en direct le taux de confiance (exprimé en %) que l'IA accorde à la détection d'un mouvement donné. Par exemple, si le taux affiché est de 50%, l'IA estime qu'il y a 1 chance sur 2 que le mouvement observé correspond à celui qu'on lui a appris. L'utilisateur peut poursuivre et enrichir l'apprentissage des mouvements afin d'augmenter les performances de la détection et se rapprocher d'un taux de 100%.

Lorsque l'utilisateur considère que l'apprentissage est suffisant, il déclenche la génération d'un bloc de programmation du type « mouvement x détecté » qui pourra être utilisé dans un programme réalisé avec MakeCode par exemple pour compter le nombre de shoots réalisés

Observation des mouvements de la carte micro:bit sur les 3 axes X, Y, Z

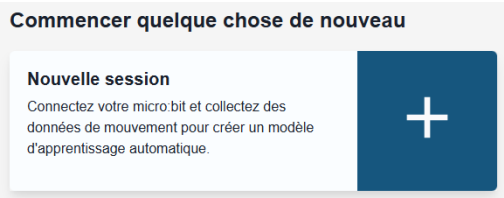
La carte micro:bit est équipée d'un capteur interne qui permet de mesurer les accélérations et décélérations de la carte selon trois axes X, Y, Z. Les axes sont orientés par rapport à la carte selon les directions indiquées ci-dessous.



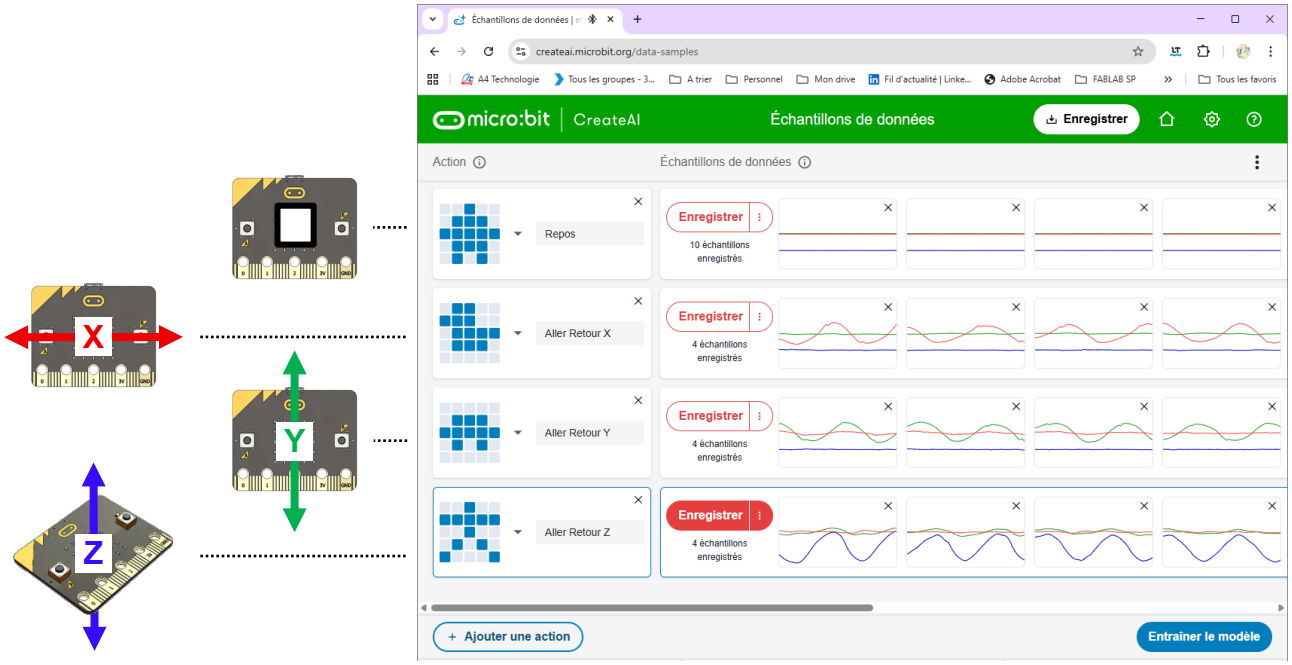
L'environnement micro:bit CreateAI permet de visualiser en direct les accélérations et décélérations de la carte selon les trois axes X, Y et Z et de créer grâce à une IA un modèle d'entraînement qui permet de créer des blocs spécifiques qui seront utilisés dans un programme réalisé avec MakeCode.

L'exemple suivant illustre les différentes possibilités de différencier des mouvements rectilignes de la carte micro:bit selon les 3 directions X, Y, Z.

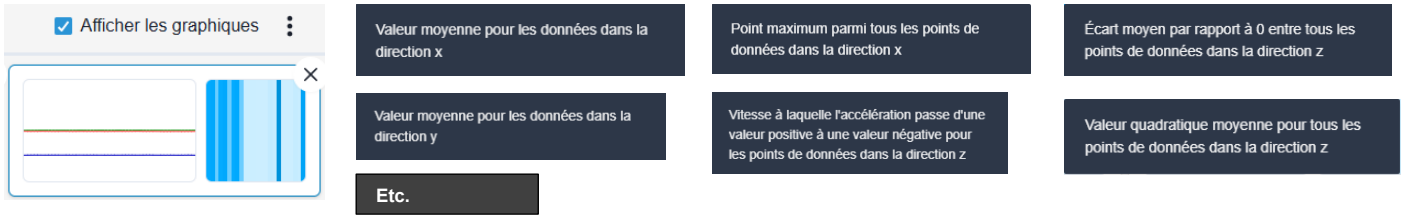
1) Lancer une nouvelle cession de micro :bit CreateAI et suivre attentivement les instructions de mise en service qui s'affichent au fur et à mesure à l'écran.



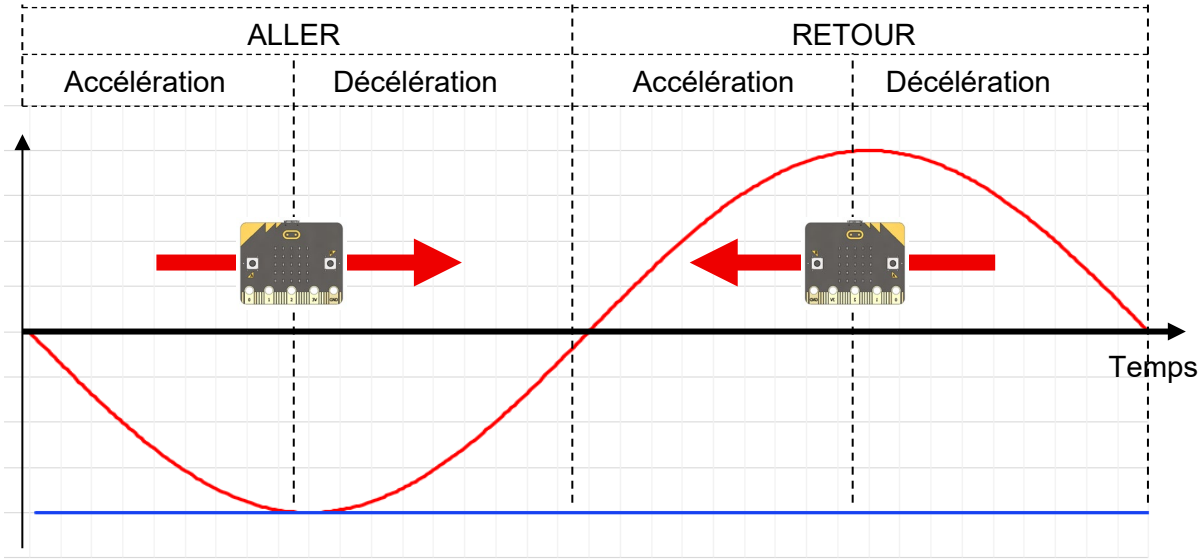
- Disposer la carte à plat sur une table et entraîner l'IA à détecter des mouvements d'allers et retours selon les 3 axes.
- Enregistrer les différentes actions « Repos », « Aller Retour X », « Aller Retour Y », « Aller Retour Z »
- Débuter les mouvements répétitifs avant de lancer l'enregistrement
- Noter qu'il est possible d'utiliser les options d'enregistrement de « 10 échantillons à la suite » ou « Enregistrer pendant 10 secondes »



Les courbes rouges, vertes et bleues, représentent les alternances des mouvements d'aller et retour de la carte. L'affichage des fonctionnalités des données permet de visualiser au survol le type de formule mathématique appliqué.

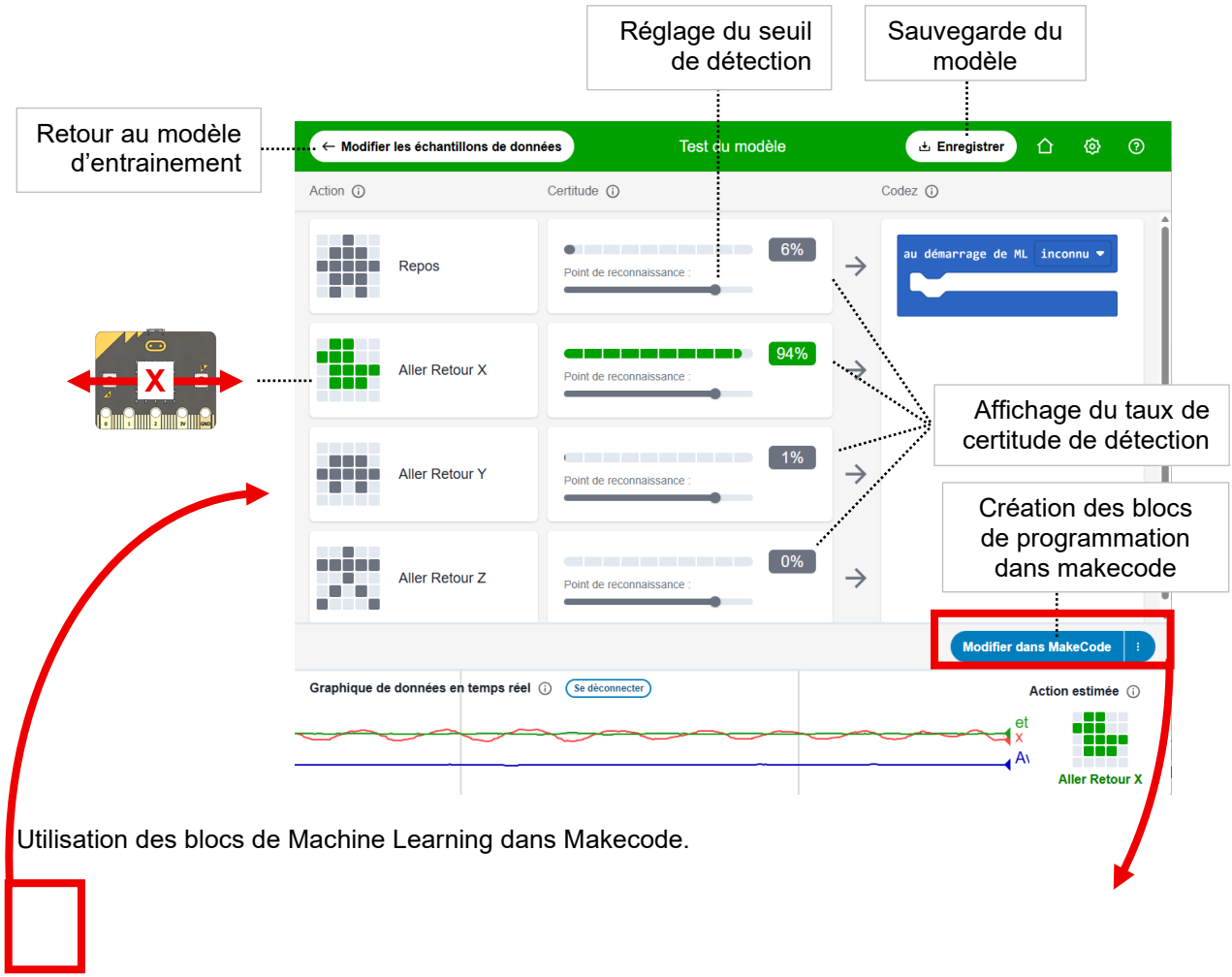


Détail de la courbe d'accélération et de décélération sur l'axe X

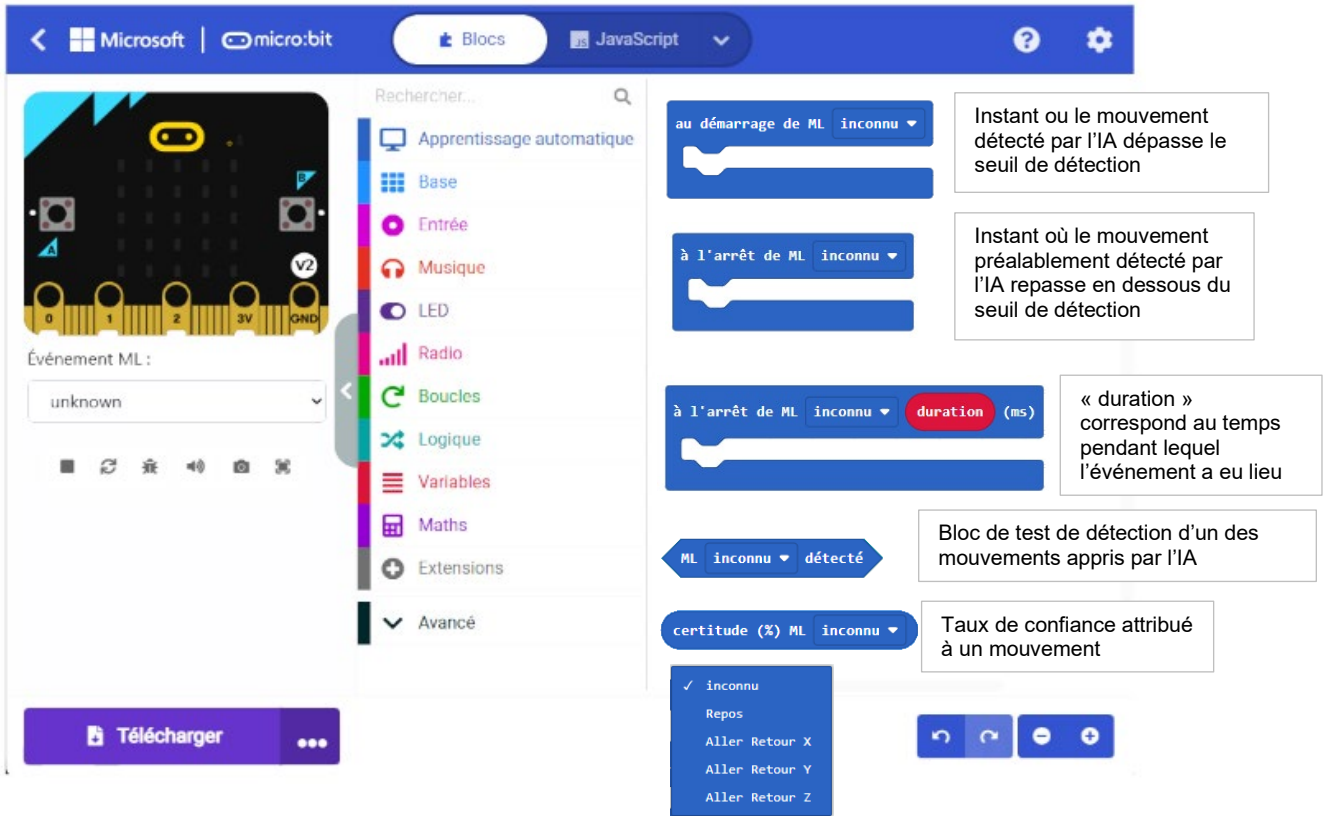


Vérification du modèle d'entraînement

Le test du modèle d'entraînement permet de vérifier que chaque mouvement est détecté avec suffisamment de certitude par l'IA. Si ce n'est pas le cas, il est nécessaire de revenir en arrière pour enrichir l'entraînement du modèle avec des données supplémentaires et d'ajuster éventuellement le seuil de détection des différents événements.



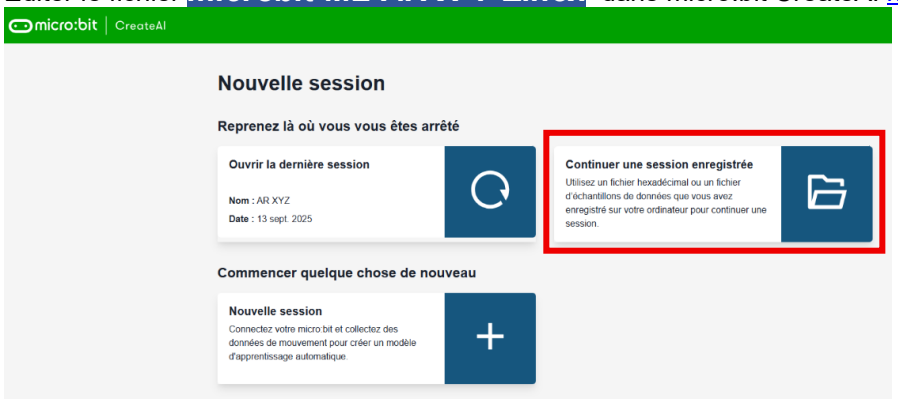
Utilisation des blocs de Machine Learning dans Makecode.



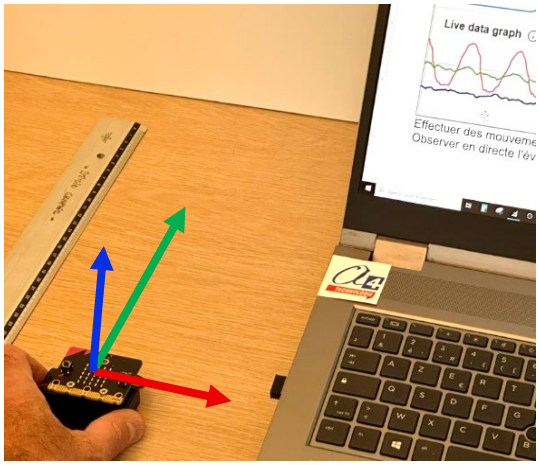
Observation des mouvements de la carte micro:bit

Les exemples de programmes suivants illustrent l'utilisation des différents blocs générés par l'IA pour détecter les mouvements alternés de la carte micro:bit selon les axes X, Y, Z.

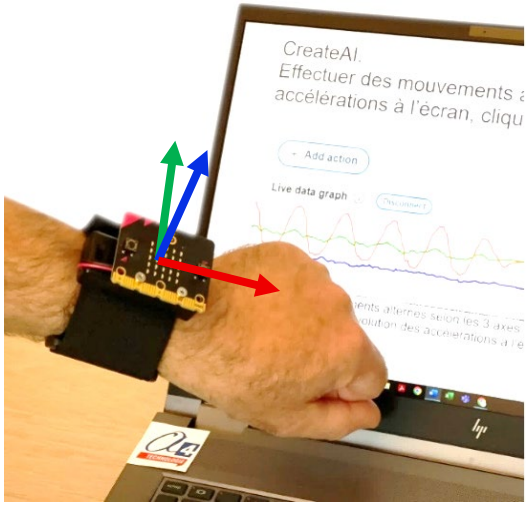
Éditer le fichier ***microbit-ML-AR-X-Y-Z.hex*** dans micro:bit CreateAI https://createai.microbit.org/new



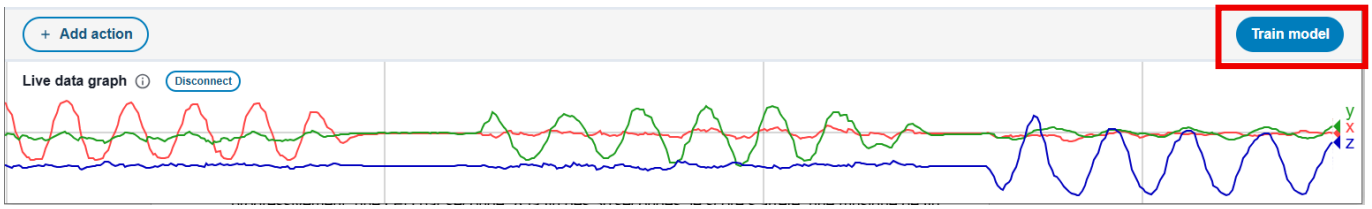
Ce fichier a été conçu en enregistrant des séquences de mouvements en posant la carte micro:bit à plat sur une table afin de faciliter son déplacement dans une seule direction à la fois.



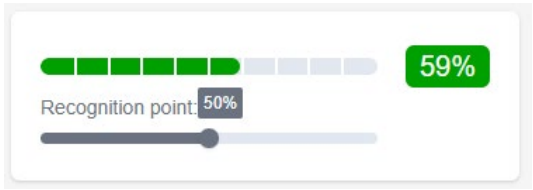
Si la carte micro:bit est portée au poignet, il est plus compliqué des répéter des mouvements identiques, car son orientation dans l'espace risque de varier d'un mouvement à l'autre. Il est possible d'enrichir ce modèle d'entraînement afin de s'adapter à un autre contexte de mouvements de la carte.



Connecter la carte micro:bit V2 à l'ordinateur et suivre les instructions pour établir la connexion avec micro:bit CreateAI.
Effectuer des mouvements alternés de la carte selon les 3 axes X, Y, Z pour observer en direct l'évolution des accélérations à l'écran, cliquer sur « Train model » pour générer le modèle d'entraînement

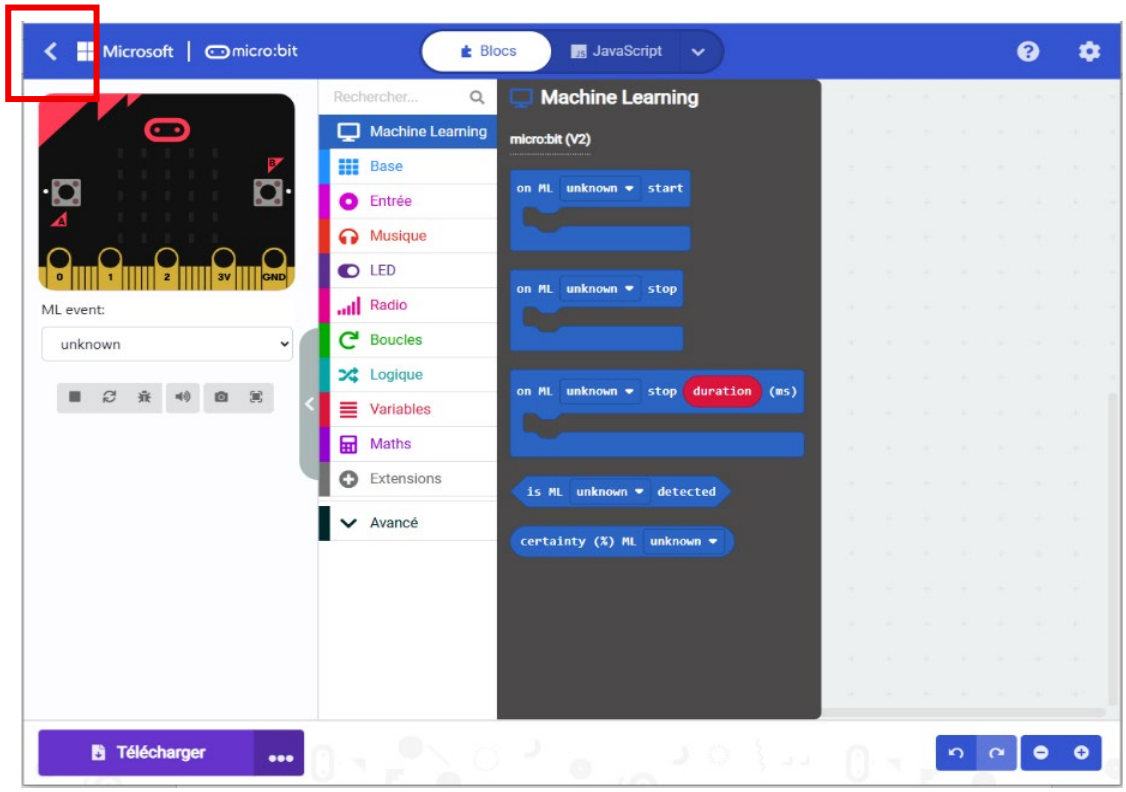


Ajuster éventuellement les seuils de détection souhaités pour considérer qu'un mouvement donné est détecté ou non



Cliquer sur « Edit in MakeCode pour que l'IA génère les blocs de programmation correspondant au modèle d'entraînement en cours.

Si nécessaire, revenir aux étapes précédentes en cliquant sur le symbole < en haut à gauche de la fenêtre



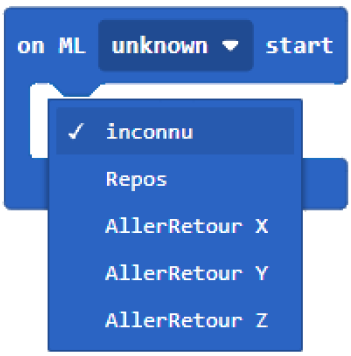
.

Fonctionnement des blocs de machine learning dans MakeCode

POINT D'ATTENTION :
L'affichage sur la matrice LED ou la génération de sons mobilisent la carte micro:bit et peuvent ralentir significativement la détection de mouvements appris. Pour augmenter la réactivité de détection de mouvements appris, il est conseillé de faire appel aux fonctions d'affichage ou de génération de sons de manière ponctuelle en donnant la priorité aux traitements de détection de mouvements.

Bloc [on ML - - - start]

Ce bloc permet de surveiller en permanence les évènements de la liste et de déclencher une action (affichage, son, émission radio, calcul, etc.) dès qu'ils débutent. L'évènement dont le taux de confiance est le plus élevé et supérieur au seuil de détection paramétré lors de l'apprentissage l'emporte sur tous les autres. Le premier élément de la liste « inconnu », permet de définir l'action à réaliser si un événement non appris est détecté.

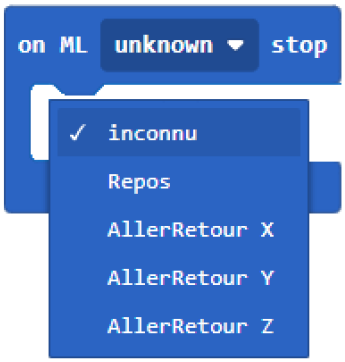


Charger le programme *TEST AR XYZ O-N ML START.hex*

Déplacer la carte en faisant des allers et retours selon les 3 axes X, Y et Z. Vérifier si les « R », « X », « Y » et « Z » sont cohérents avec les déplacements effectués.
Observer les situations où l'information « I » correspondant à un mouvement non appris survient, en vue d'améliorer le modèle d'apprentissage.

Bloc [on ML - - - stop]

Ce bloc permet de surveiller en permanence les évènements de la liste et déclencher une action dès que l'évènement en cours cesse. L'évènement dont le taux de confiance est le plus élevé et est supérieur au seuil de détection paramétré lors de l'apprentissage devient alors l'évènement en cours.



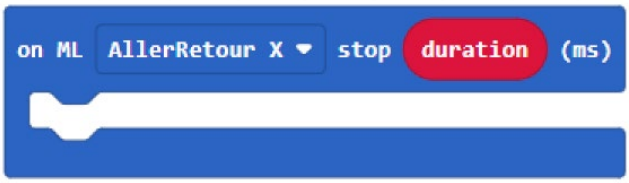
Charger le programme ***TEST AR XYZ -ON ML STOP.hex***

Déplacer la carte en faisant des allers et retours selon les 3 axes X, Y et Z. Vérifier si les « R », « X », « Y » et « Z » sont cohérents avec les déplacements effectués.
Observer les situations où l'information « I » correspondant à un mouvement non appris survient, en vue d'améliorer le modèle d'apprentissage.

Points d'attention :
L'utilisation de l'affichage sur la matrice LED peut ralentir significativement les temps de traitements et de détection des évènements.
Si les mouvements sont enchaînés, on remarque un temps de latence de l'affichage.

Bloc [on ML - - - stop (duration) ms]

Ce bloc permet de mesurer la durée de l'événement en cours. L'évènement en cours est celui dont le taux de confiance est le plus élevé et est supérieur au seuil de détection paramétré lors de l'apprentissage. Lorsque cet évènement cesse, la variable « duration » est renseignée avec le temps (exprimé en millisecondes) pendant lequel cet évènement a été actif.

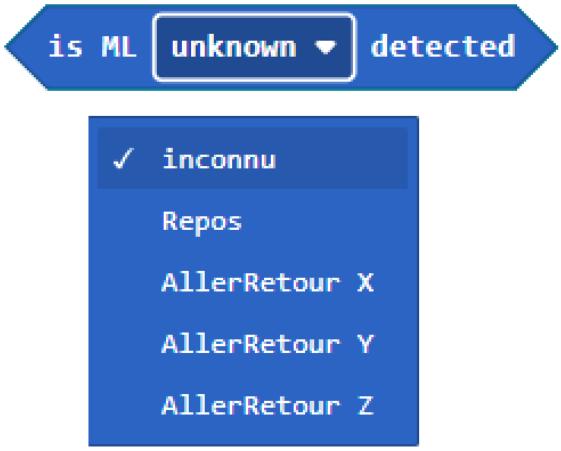


Charger le programme *TEST-ON-ML-STOP-duration.hex*

Déclencher un chronomètre et déplacer la carte pendant quelques secondes en faisant des allers et retours selon l'axe X. Arrêter le chronomètre et arrêter de déplacer la carte. Vérifier que le temps affiché sur la carte est identique à celle affichée sur le chronomètre ;
La durée affichée sur la carte est exprimée en secondes pour faciliter la lecture.

Bloc [is ML - - - detected]

Ce bloc peut être utilisé dans un test conditionnel pour déterminer quel déplacement est détecté.

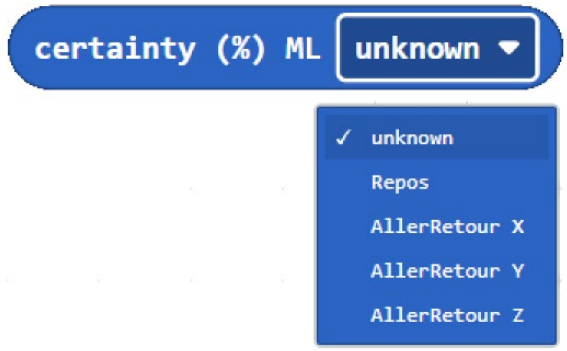


Charger le *TEST-IS-ML-DETECTED.hex*

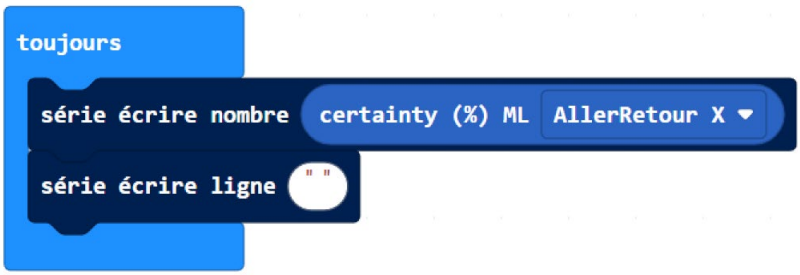
Déplacer la carte en faisant des allers et retours selon les 3 axes X, Y et Z. Vérifier si les « R », « X », « Y » et « Z » sont cohérents avec les déplacements effectués.
Observer les situations où l'information « I » correspondant à un mouvement non appris survient, en vue d'améliorer le modèle d'apprentissage.

Bloc [certainly (%) ML …]

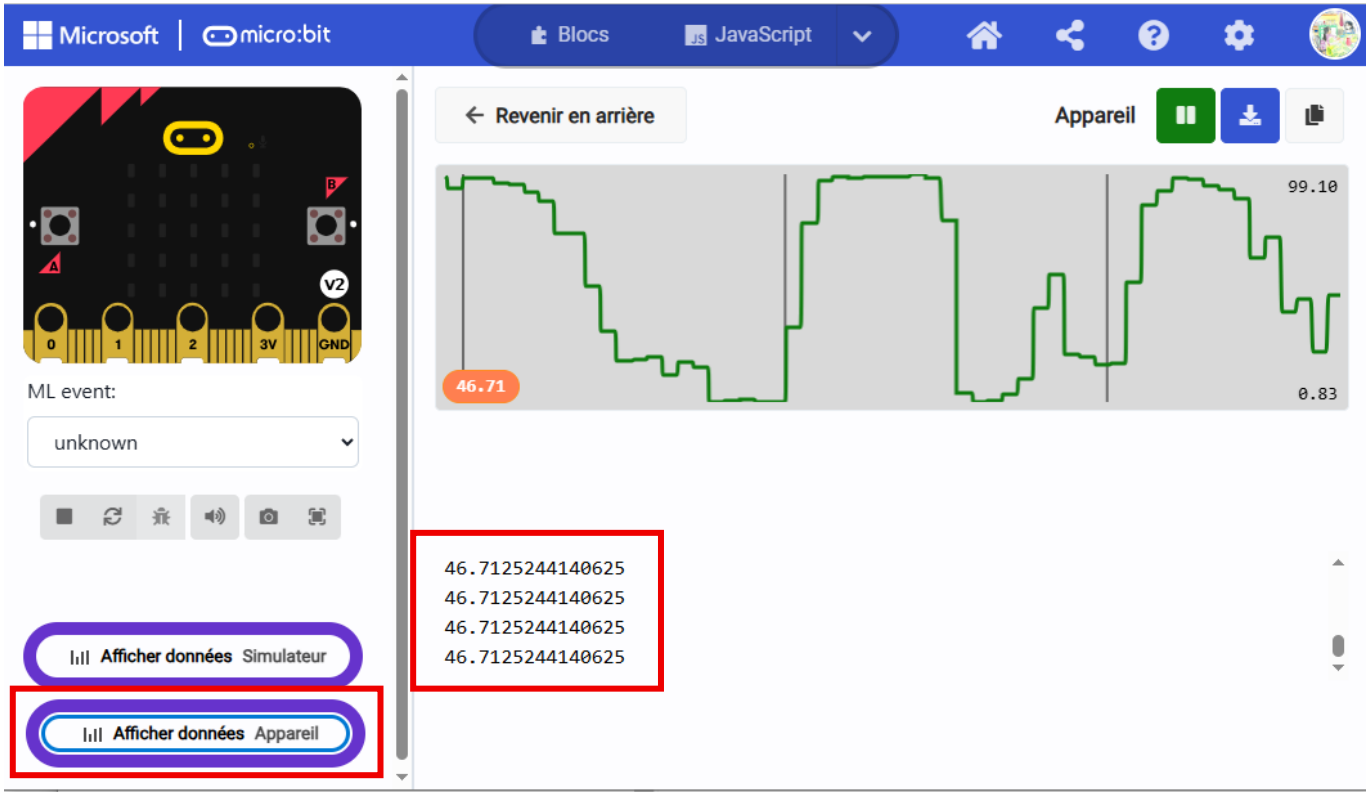
Ce bloc collecte le taux de confiance (en %) accordé à la détection d'un évènement.



Charger le programme *TEST-CERTAINLY-ML.hex*.

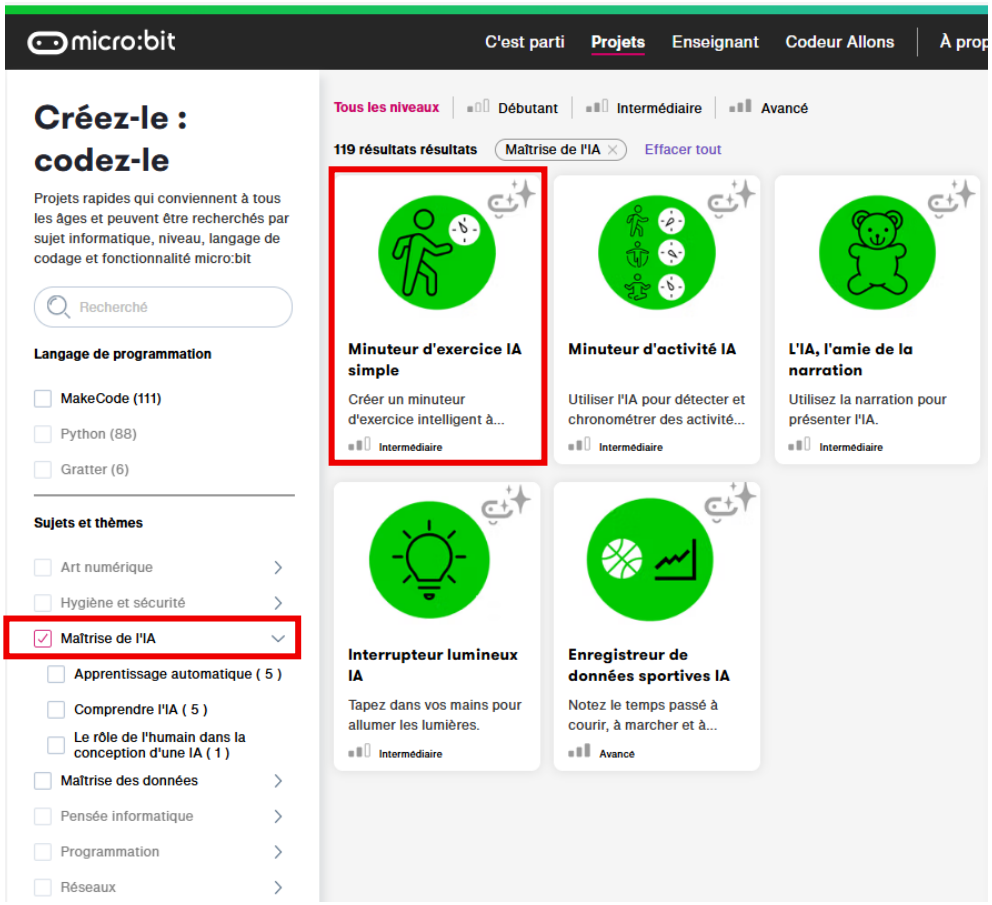


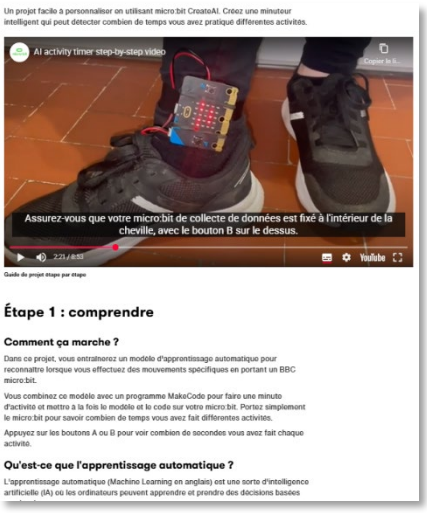
Afficher les données de l'appareil.
Déplacer la carte en faisant des allers et retours sur l'axe X et observer les valeurs retournées.

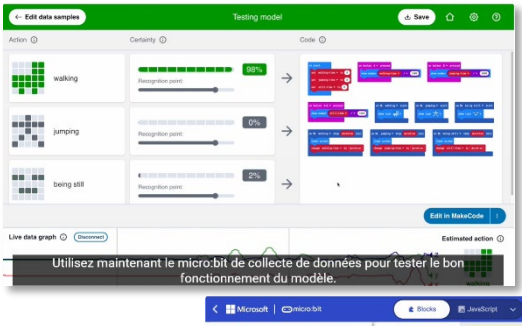


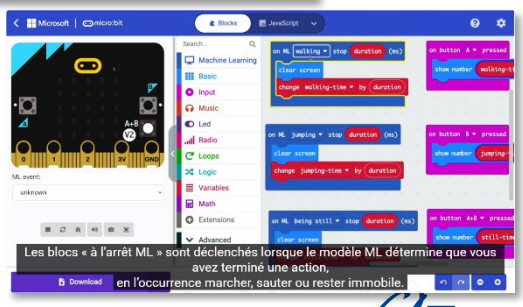
Tutoriels et projets IA avec micro:bit

https://microbit.org/fr/projects/make-it-code-it/?filters=bd4c25b7-652b-4d0d-94c8-9815d8f32cf4_fr









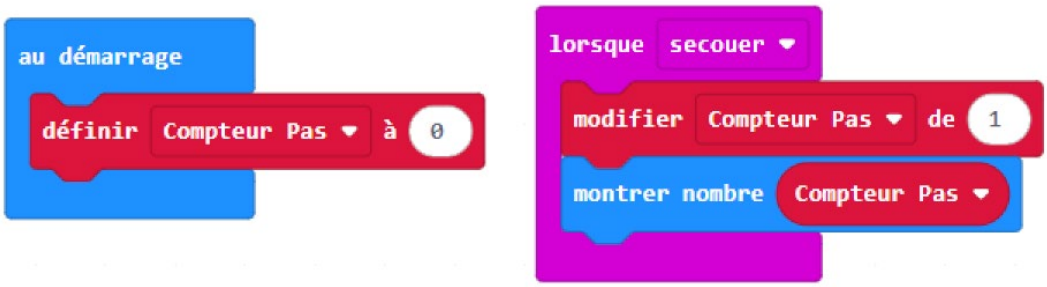
Activités de programmation avec microMOUV

Compteur de pas simple

But du programme :
Utiliser le bracelet microMOUV fixé au poignet afin de réaliser un compteur de pas.
Exploiter les possibilités offertes par les blocs de programmation destinés à détecter les mouvements de la carte.

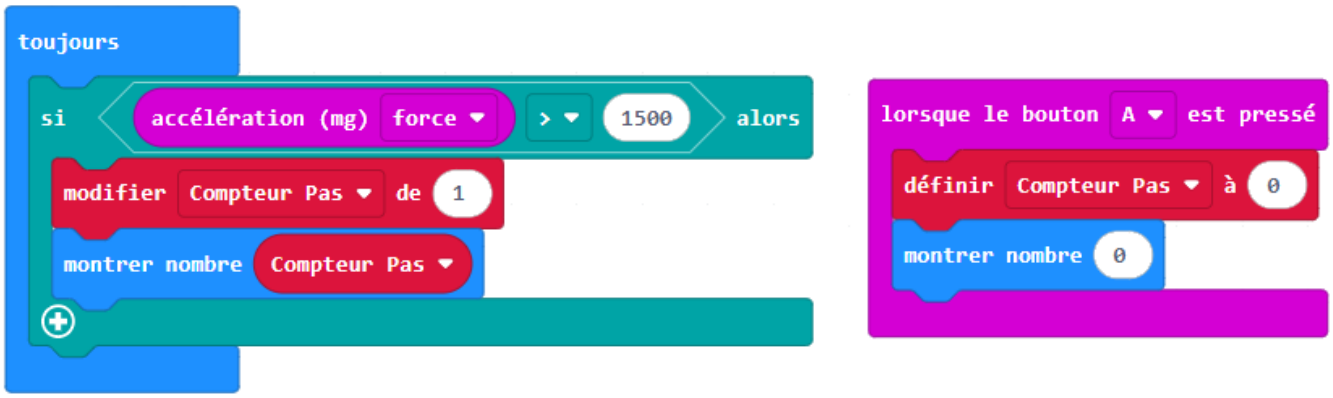
Blocs de programmation disponibles :

Exemple de programme ***microbit-Compteur-de-pas.hex***
Lorsqu'une l'accélération dépasse un certain seuil, la variable « Compteur de pas » est incrémentée puis est affichée. L'appui sur le bouton A remet le compteur à zéro.



Améliorations du programme :
Surveiller si les accélérations de la carte dépassent un seuil donné. Utiliser le bouton A pour remettre à zéro le compteur.

Exemple de programme *microbit-Compteur-de-pas-V2.hex*

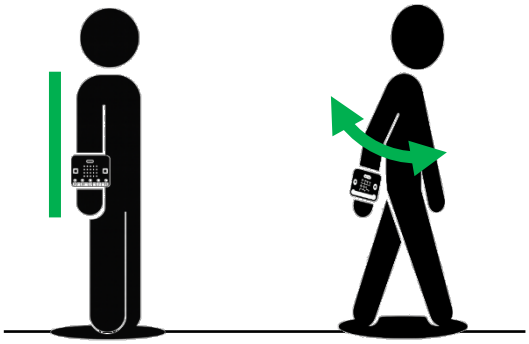


Ajuster éventuellement la valeur affichée afin de la faire correspondre au nombre réel de pas

Compteur de pas par apprentissage IA

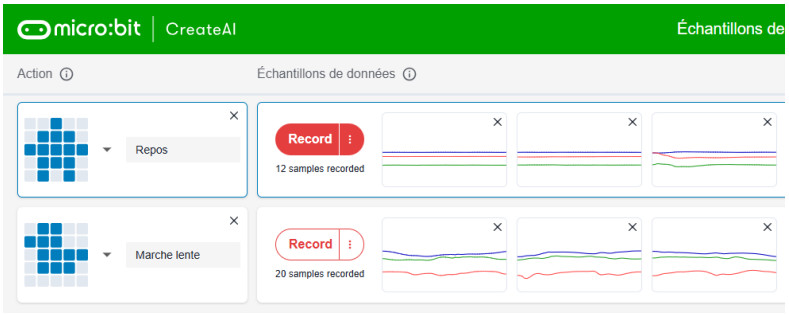
But du programme :
Utiliser le bracelet microMOUV fixé au poignet afin de réaliser un compteur de pas.
Utiliser micro:bit Create AI pour générer des blocs de programmation générés par apprentissage du mouvement du bras pendant la marche.

Conditions d'entraînement du modèle IA :
Le bracelet microMOUV est fixé au poignet droit avec le symbole micro:bit orienté vers l'extérieur.
Deux situations d'entraînement sont prises en compte :



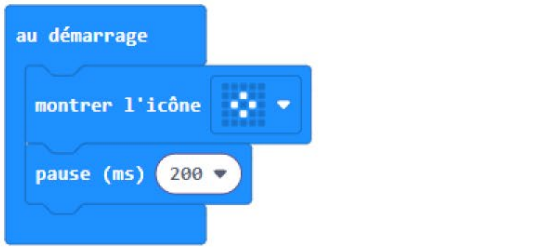
Repos
Bras le long du corps

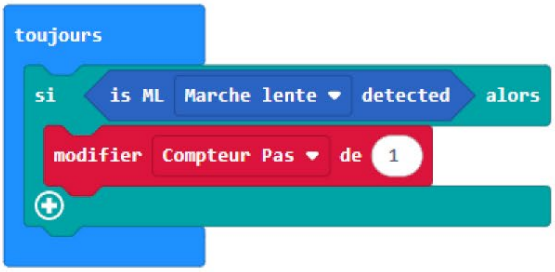
Marche lente
Léger balancement naturel des bras



Modèle d'entrainement

Exemple de programme : *ML-Compteur-de-pas.hex*
Le compteur de pas est incrémenté en permanence dès que la marchez est détectée. On applique un calcul pour retranscrire le nombre de pas réel réalisés.

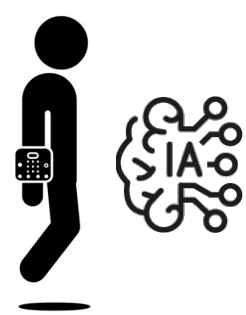




Améliorations du programme et du modèle d'entraînement :
Comparer le nombre de pas effectués avec le nombre de pas comptés par le programme.
Ajuster la valeur du dénominateur dans la formule de calcul en fonction de la foulée de l'utilisateur.

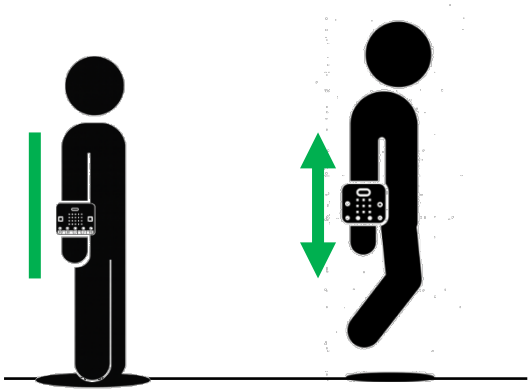
Vérifier le nombre de pas réalisés en portant le bracelet microMOUV au poignet gauche.
Enrichir le modèle d'entraînement afin de prendre en compte ce cas.

Compteur de saut par apprentissage IA



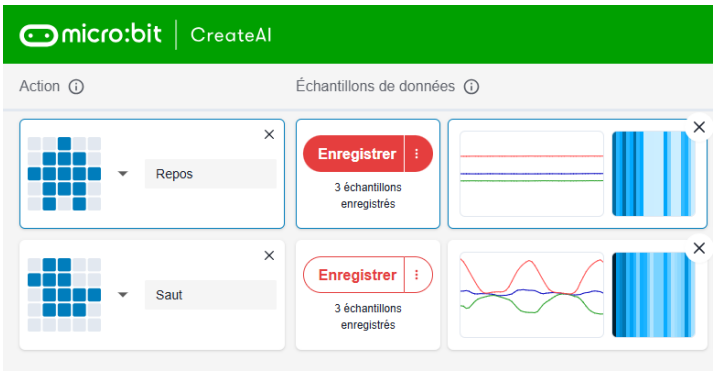
But du programme :
Programmer le bracelet microMOUV afin de réaliser un compteur de sauts.
Utiliser l'environnement micro:bit Create AI pour créer des blocs sur mesure

Conditions d'entraînement du modèle IA :
Le bracelet microMOUV est fixé au poignet gauche avec le symbole micro:bit orienté vers l'extérieur. Deux situations d'entraînement sont prises en compte :

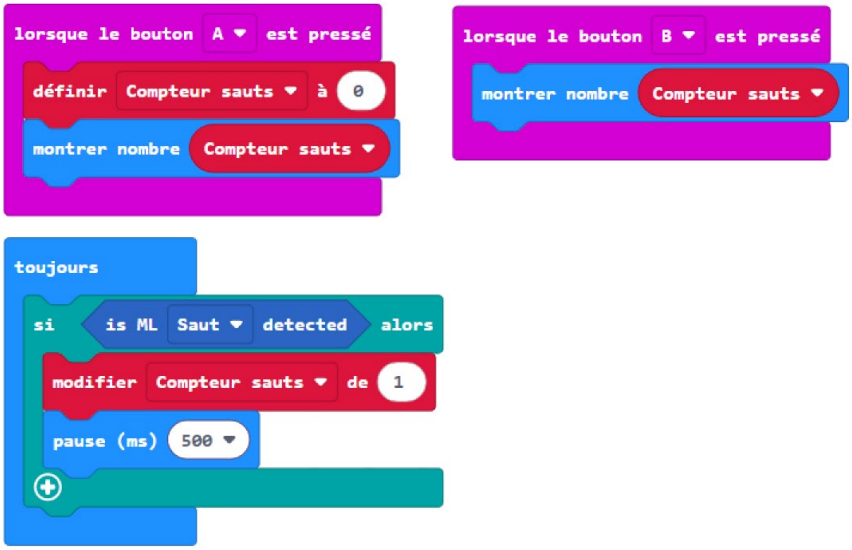


Repos
Bras le long du corps

Saut sur place
Bras le long du corps



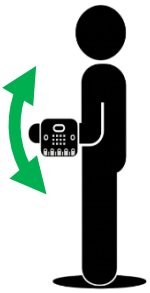
Exemple de programme : *ML-Compteur saut V1*
Lorsqu'une séquence de sauts les bras le long du corps est détectée, la variable « Compteur sauts » est incrémentée toutes les 500 ms. Le bouton A permet de remettre à zéro la variable et le bouton B à l'afficher.



Améliorations du programme et du modèle d'entraînement :
Comparer le nombre de sauts effectués avec le nombre de sauts comptés par le programme. Si nécessaire, ajuster le temps d'attente afin d'obtenir un comptage cohérent avec la réalité.

Plier le bras gauche en restant immobile et vérifier le comptage. Enrichir le modèle d'entraînement afin de prendre en compte les situations aberrantes prises pour des sauts.

Enrichir le modèle d'entraînement en portant le bracelet sur l'autre bras (bras droit)

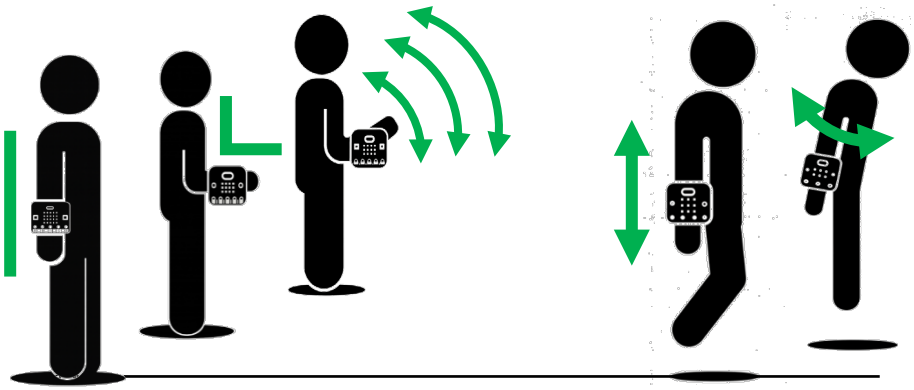


Chrono Jump

But du programme :
Programmer le bracelet microMOUV afin d'afficher la durée totale d'une série ininterrompue de sauts sur place. L'utilisateur lance un compte à rebours de quelques secondes en appuyant sur le bouton A puis se maintient en position statique les bras le long du corps. À l'issue du décompte un « bip » long est émis pour lui indiquer de commencer sa série de sauts. Dès que la série de sauts est interrompue, un bip grave est émis et la durée de la série de sauts (exprimé en secondes) s'affiche en défilant sur la matrice de LED. L'appui sur le bouton B réactive l'affichage de la durée de la série. L'appui sur le bouton A permet de réinitialiser le bracelet pour recommencer une nouvelle série.

Conditions d'entraînement du modèle IA :
Le bracelet microMOUV est fixé au poignet droit avec le symbole micro:bit orienté vers l'extérieur. Plusieurs situations d'entraînement sont prises en compte pour différencier de manière fiable la détection de sauts sur place et les périodes de récupérations.

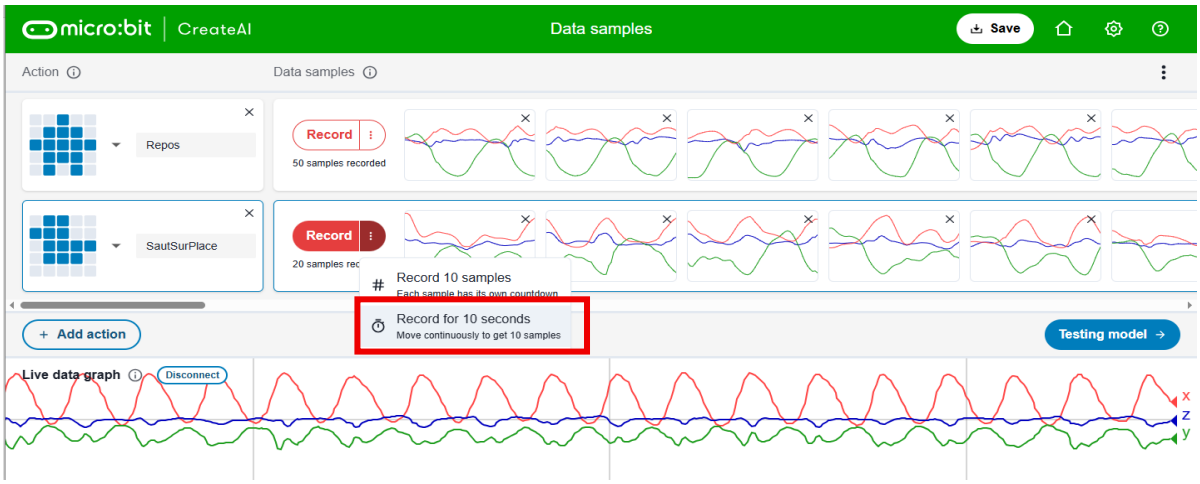
Répéter séparément les séries de gestes



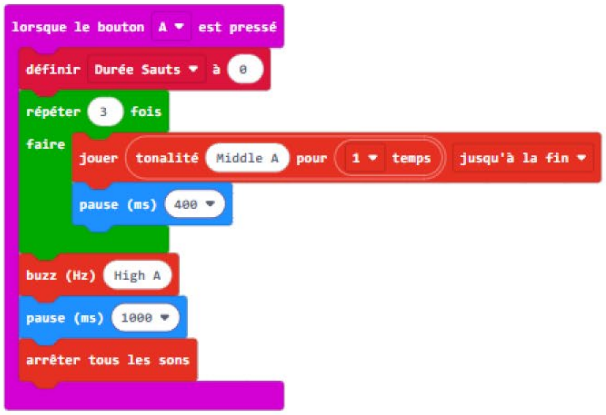
Repos
Utilisateur immobile, bras le long du corps ou avant-bras

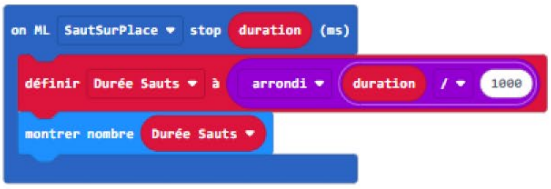
Saut sur place
Bras le long du corps ou léger balancement durant le saut

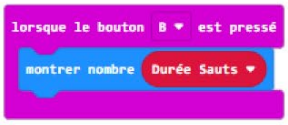
Apprentissage :
Enregistrer séparément chaque séquence de mouvements répétés en utilisant le mode d'enregistrement pendant 10 secondes. Vérifier le modèle, enrichissez-le en associant des séquences de gestes aux types de mouvements que l'IA devra reconnaître. Vérifier la fiabilité de détection des 2 types de mouvements (repos ou Saut sur place). Ajuster éventuellement les seuils de déclenchement pour fiabiliser la différenciation des 2 mouvements.

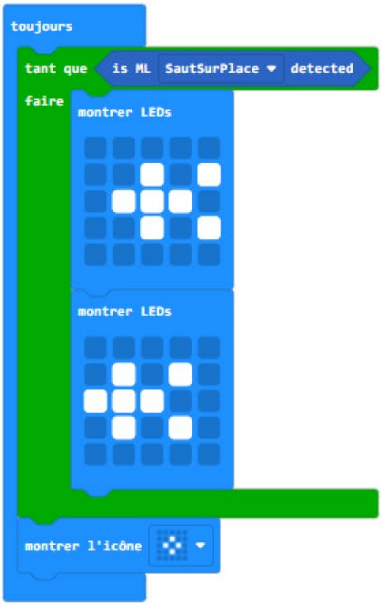


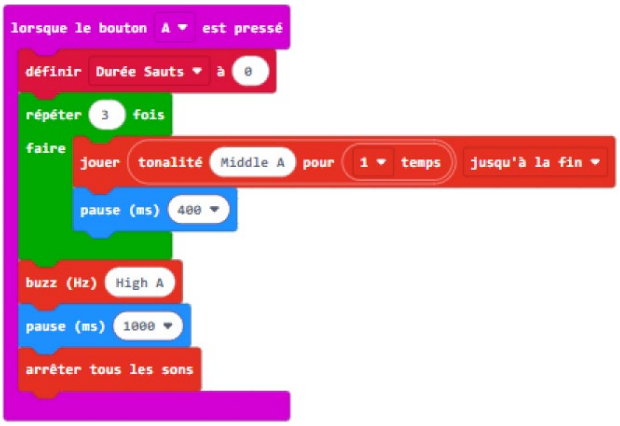
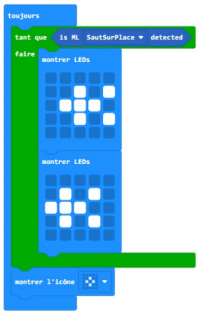
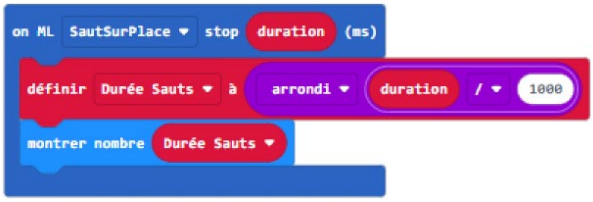
Exemple de programme : *ML-microbit-ChronoJump_V1.hex*







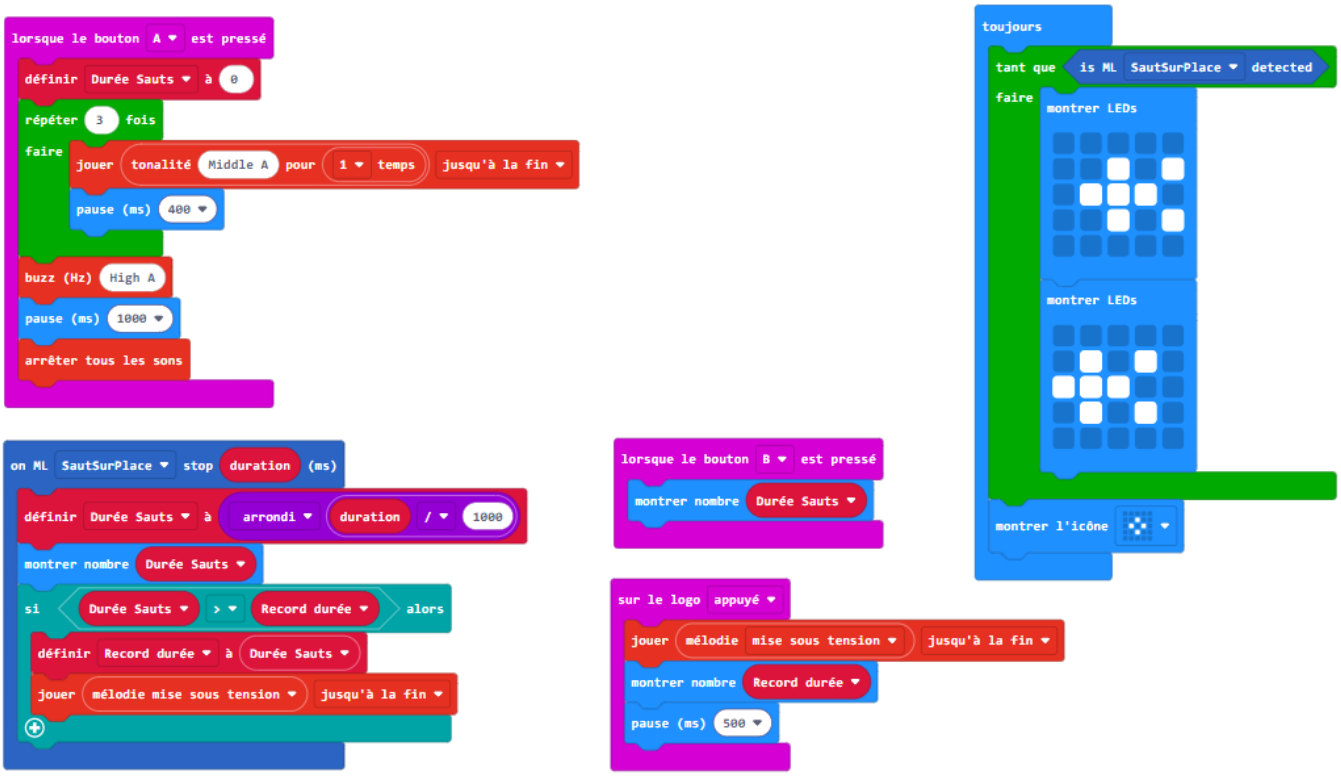


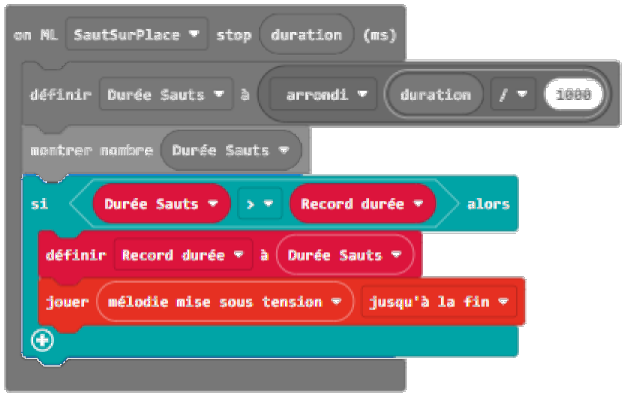
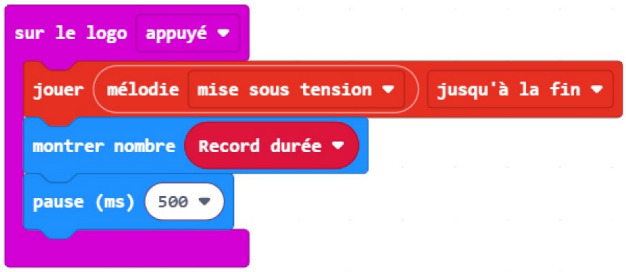
Bloc de programmation	Commentaire
	L'appui sur le bouton A permet d'initialiser le programme en remettant à la variable « Durée Sauts » et de déclencher une séquence sonore pour indiquer à l'utilisateur le moment où il peut commencer les sauts sur place
	Animation lumineuse d'un personnage qui saute lorsque l'IA détecte qu'une séquence de sauts est en cours
	La variable « duration » consigne le temps écoulé entre le début et la fin d'une séquence de sauts. La variable « Durée sauts » est mise à jour dès qu'une séquence de sauts prend fin. Le calcul effectué permet de convertir les ms en secondes et d'arrondir le résultat.
	Le bouton B permet de lancer l'affichage du temps consacré à la dernière série de sauts.

Jump champion

Enrichir le programme précédent (Chrono Jump V1) afin de mémoriser le temps le plus long réalisé lors de séquences successives de sauts. Afficher ce temps à l'appui du bouton tactile et lancer une mélodie.

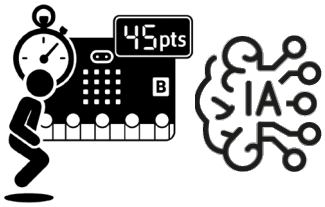
Exemple de programme : *ML-microbit-ChronoJump_V2.hex*



Bloc de programmation	Commentaire
	La durée de sauts la plus longue est mémorisée dans la variable « Record durée » à l'issue de chaque séquence de sauts.
	Le bouton tactile permet de lancer une mélodie et d'afficher le temps record de durée

Jump Trainer

But du programme :
Configurer le bracelet microMOUV comme assistant d'entraînement pour améliorer ses performances lors de sessions chronométrées de sauts sur place.
Un compte à rebours de 1 minute est lancé au début de la séquence. Tout au long de l'exercice, le bracelet comptabilise des points en fonction de la fréquence des sauts :
- Aucun point si l'utilisateur est immobile ou qu'il saute pendant des intervalles de temps inférieurs à 2 secondes,
- 1 point toutes les 3 secondes si l'utilisateur saute à vitesse lente,
- 2 point toutes les 3 secondes si l'utilisateur saute à vitesse rapide.



À l'issue du compte à rebours, le score s'affiche sur la matrice de LED (cumul des points collectés pendant 1 min).

L'appui sur le bouton A relance le compte à rebours pour une nouvelle cession, l'appui sur le bouton B provoque l'affichage du score de la cession.

Décomposition du problème :

Les étapes suivantes correspondent à des parties du programme qu'il est souhaitable de maîtriser individuellement avant de les assembler pour constituer le programme complet.

1. Entraîner le modèle IA pour détecter 3 situations :
   - Repos,
   - Fréquence de sauts supérieurs à un seuil F1,
   - Fréquence de sauts supérieure à un seuil F2.
2. Vérifier le bon fonctionnement du modèle d'entraînement IA :
   - Programmer la carte micro:bit pour observer les actions détectées
   - Enrichir le modèle dans micro:bit Create AI
3. Surveiller et détecter la fréquence des sauts :
   - Repos,
   - Fréquence de sauts supérieure à un seuil F1 pendant plus de 1,5 seconde
   - Fréquence de sauts supérieure à un seuil F2 pendant plus de 1,5 seconde
4. Déclencher et gérer un compte à rebours de 1 minute à l'appui du bouton A
5. Gérer le comptage des points :
   - Repos = 0 point
   - Fréquence >= F1 et <F2 ➔ +1 point toutes les 3 secondes
   - Fréquence >= F2 ➔ +2 points toutes les 3 secondes
6. Afficher le score à l'issue du compte à rebours à l'appui du bouton B

1

Entraîner le modèle IA pour détecter 3 situations :
- Repos,
- Fréquence de sauts supérieurs à un seuil F1,
- Fréquence de sauts supérieure à un seuil F2.

Exemple de programme : ML-microbit-Jump-Trainer-1.hex

Glisser / Déposer le fichier exemple directement dans [micro:bit CreateAI](#).

Note : les données d'entraînement de ce modèle ont été collectées avec le bracelet microMOUV porté sur le poignet gauche.

Repos : l'utilisateur est statique, il ne saute pas, ses bras pendent le long du corps
Saut F1 : l'utilisateur fait des sauts à faible hauteur. La courbe rouge met en évidence qu'il réalise au moins 2 sauts par intervalle de temps.
Saut F2 : l'utilisateur fait des sauts plus haut. La courbe rouge met en évidence qu'il reste plus longtemps en l'air.

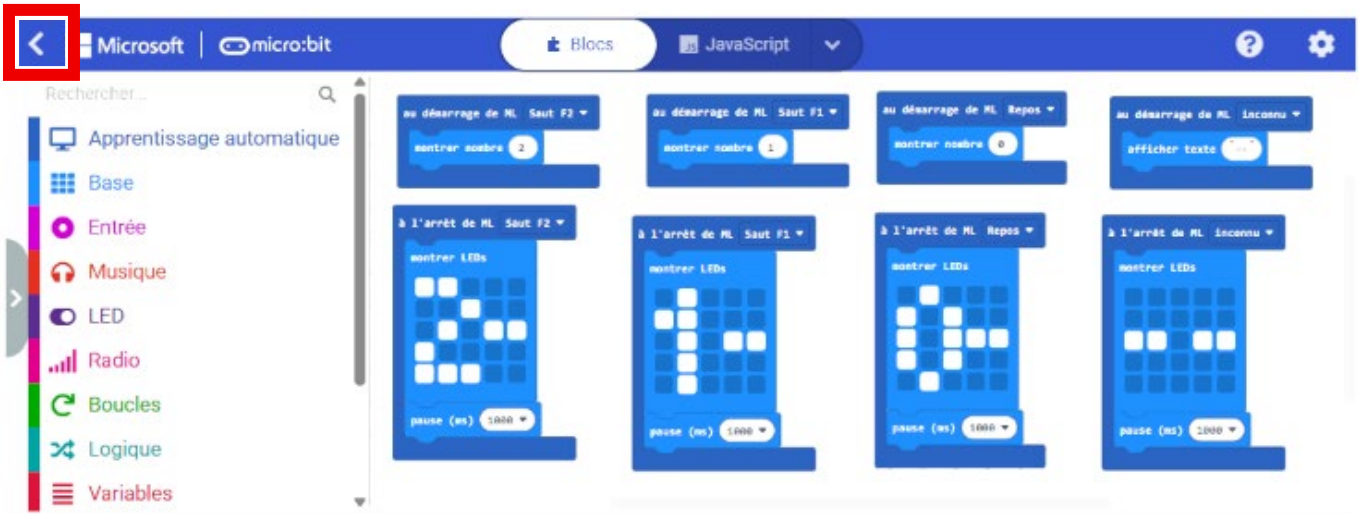
2 Vérifier le bon fonctionnement du modèle d'entraînement IA :
- Programmer la carte micro:bit pour observer les actions détectées
- Enrichir le modèle dans micro:bit Create AI

Effectuer des sauts à différentes cadences, observer l'affichage sur la carte pendant les périodes de sauts puis pendant les périodes de repos. Les valeurs affichées permettent de distinguer le début et la fin des différentes actions détectées.

Éditer le modèle dans Makecode, programmer la carte micro:bit en utilisant les blocs de Machine Learning générés par l'IA, utiliser la matrice de LED de la carte pour vérifier que les différents mouvements appris sont correctement détectés.

Exemple de programme : ML-microbit-Jump-Trainer-2.hex

Si nécessaire, revenir dans micro:bit CreateAI pour enrichir le modèle d'apprentissage et/ou ajuster les seuils de déclenchement afin d'optimiser la bonne détection des actions détectées par l'IA.



POINT D'ATTENTION IMPORTANT :

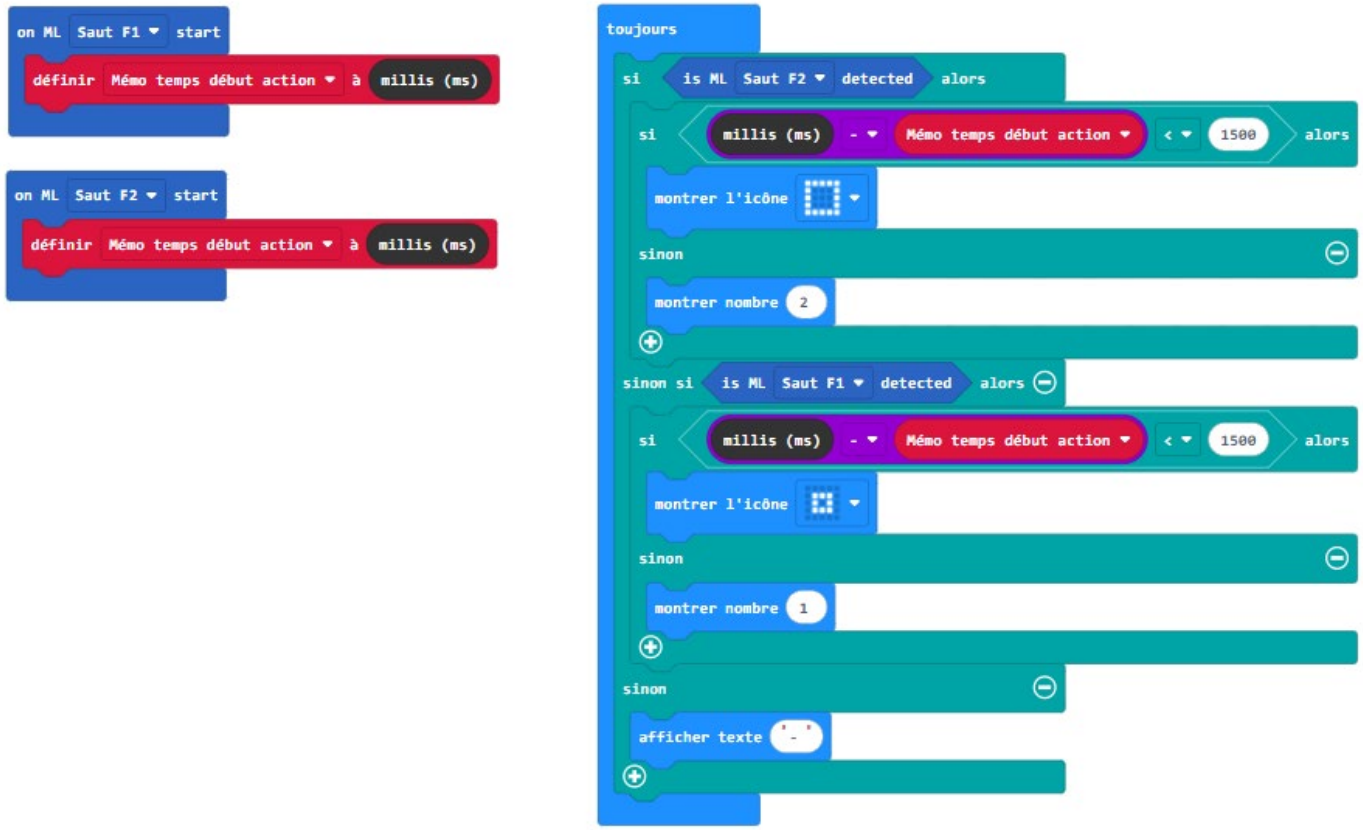
L'affichage d'informations sur la matrice de LED de la carte micro:bit requiert un temps de traitement qui peut ralentir de manière significative le processus de détection des mouvements appris. Une fois que le programme a été testé, il est recommandé de limiter l'usage de l'affichage afin d'augmenter les performances de détection de mouvements appris.

3 Surveiller et détecter la fréquence des sauts :

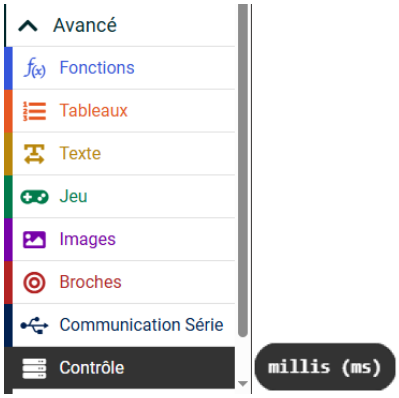
- Repos,
- Fréquence de sauts supérieure à un seuil F1 pendant plus de 1,5 seconde
- Fréquence de sauts supérieure à un seuil F2 pendant plus de 1,5 seconde

Exemple de programme : ML-microbit-Jump-Trainer-3.hex

Glisser / Déposer le fichier exemple directement dans [micro:bit CreateAI](), éditer le programme dans MakeCode puis le télécharger dans la carte micro:bit.



La carte micro:bit dispose d'une horloge interne qui fonctionne en tâche de fond. Le bloc « millis (ms) » est accessible à partir du menu « Avancé », « Contrôle ».Ce bloc correspond au nombre de millisecondes écoulées depuis le démarrage de la carte, depuis sa mise sous tension ou sa réinitialisation avec son bouton RESET.

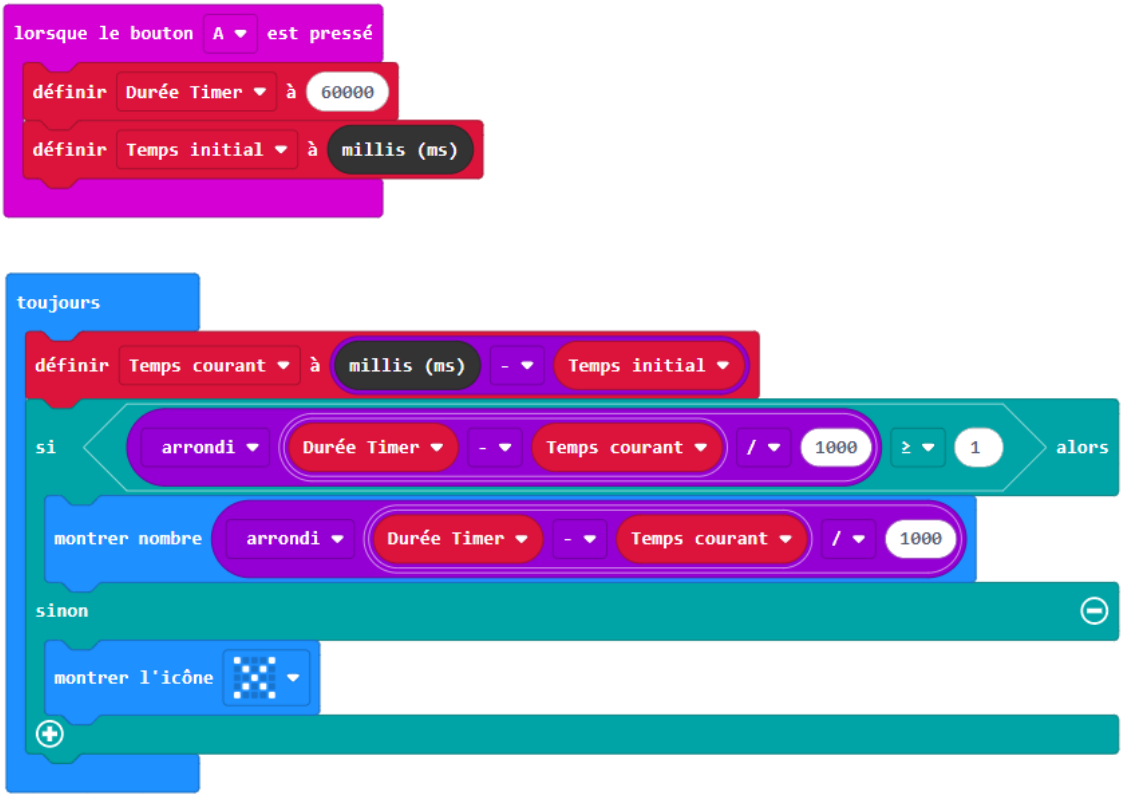


La variable « Mémo temps début action » stocke le temps de départ des sauts à la fréquence F1 ou F2 et permet de surveiller si le temps écoulé depuis le début de l'action dépasse 1500 ms (soit 1,5 s).

4 Déclencher et gérer un compte à rebours de 1 minute à l'appui du bouton A

Exemple de programme : ML-microbit-Jump-Trainer-4.hex

Glisser / Déposer le fichier exemple directement dans micro:bit CreateAI, éditer le programme dans MakeCode puis le télécharger dans la carte micro:bit.
L'appui sur le bouton A mémorise le temps courant (horloge interne de la carte micro:bit) et initialise la variable « Durée Timer » (ici initialisation à 60 000 ms soit 1minute).
La différence entre le temps courant (horloge interne de la carte) et le temps initial correspond au temps écoulé depuis l'appui sur le bouton A.
On affiche la différence entre le temps, la durée souhaitée du Timer (« Durée Timer » et le temps écoulé depuis l'appui sur le bouton A afin d'afficher un décompte du temps jusqu'à 0.
L'affichage du décompte est arrondi et exprimé en secondes.
Lorsque le décompte du temps atteint 1 seconde, on affiche une croix (X) sur la matrice de LED.

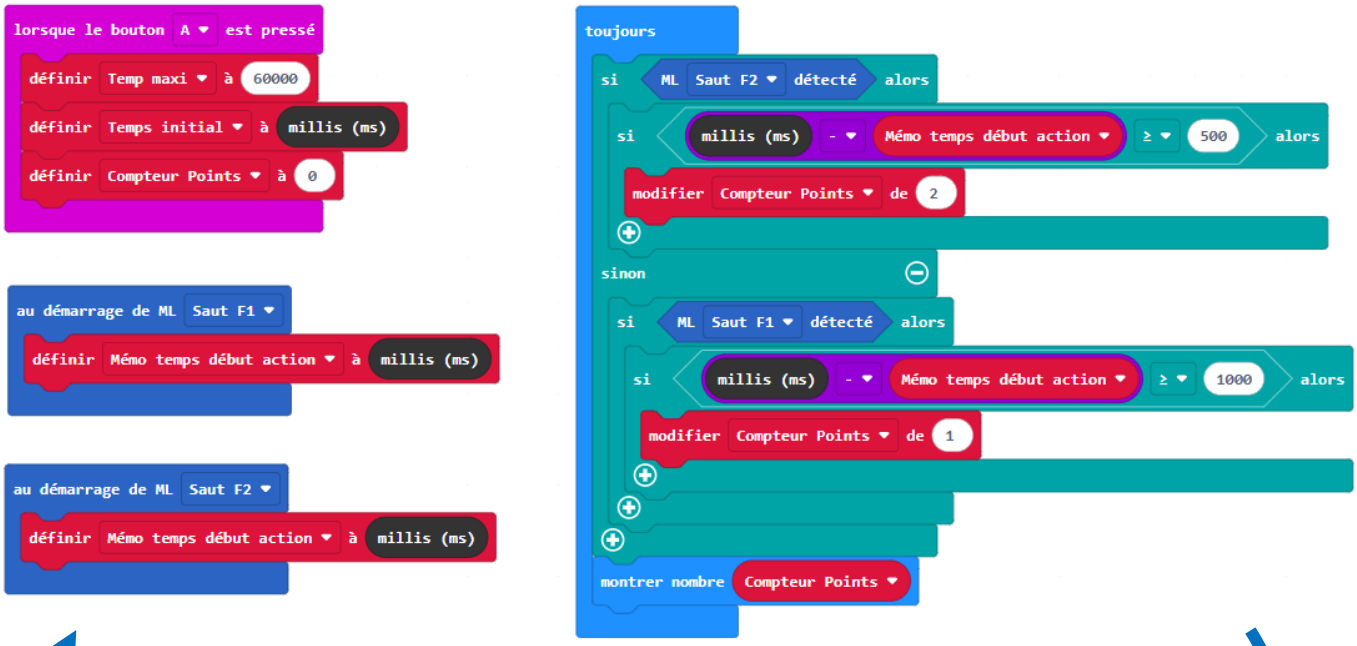


5

Gérer le comptage des points :
- Repos = 0 point
- Fréquence >= F1 et <F2 ➔ +1 point toutes les 1,5 secondes
- Fréquence >= F2 ➔ +2 points toutes les 3 secondes

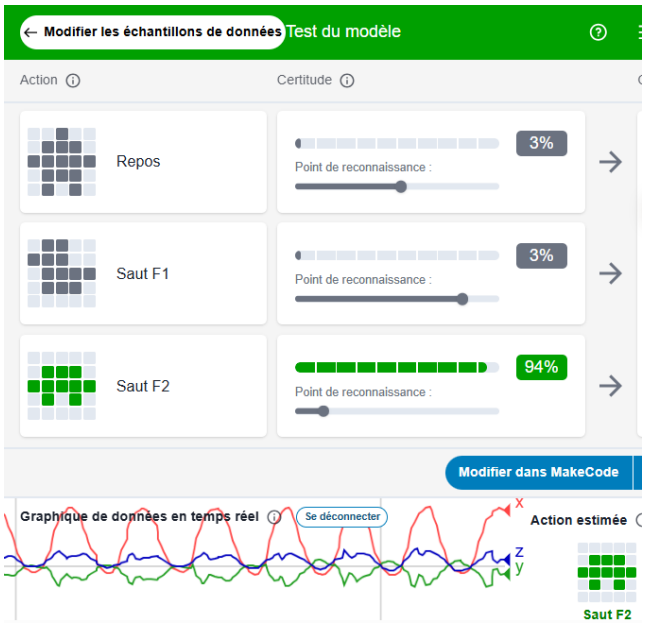
Exemple de programme : ML-microbit-Jump-Trainer-5

Glisser / Déposer le fichier exemple directement dans [micro:bit CreateAI](), éditer le programme dans MakeCode puis le télécharger dans la carte micro:bit.



TEST DU PROGRAMME ET DU MODÈLE D'ENTRAÎNEMENT

Effectuer une série de 10 sauts consécutifs à la fréquence F1 (petits sauts), relever le comptage affiché sur la carte. Faire une seconde série de 10 sauts à la fréquence F2 (grands sauts), relever le comptage affiché sur la carte pour s'assurer que le comptage est plus favorable. Ajuster les temps minimums pendant lesquels les sauts à la fréquence F2 ou F1 doivent être pris en compte, ajuster la valeur d'incrément des points, enrichissez le modèle d'entraînement, ajustez les seuils de détection des différentes actions.



6

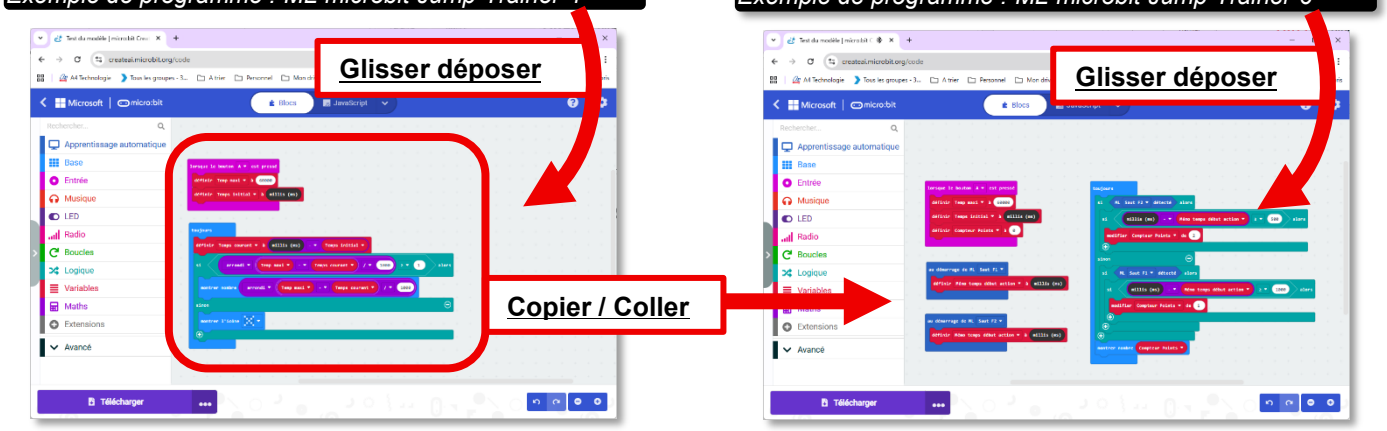
Afficher le score à l'issue du compte à rebours à l'appui du bouton B

Exemple de programme : ML-microbit-Jump-Trainer-6

ASTUCE : lancer micro:bit CreateAI, dans 2 fenêtres de navigation distincte, glisser la dernière version du programme dans une fenêtre et la version ayant servi pour mettre au point la gestion du compte à rebours dans une autre fenêtre. Utiliser le clic droit de la souris pour copier / coller les blocs souhaités dans la version finale.

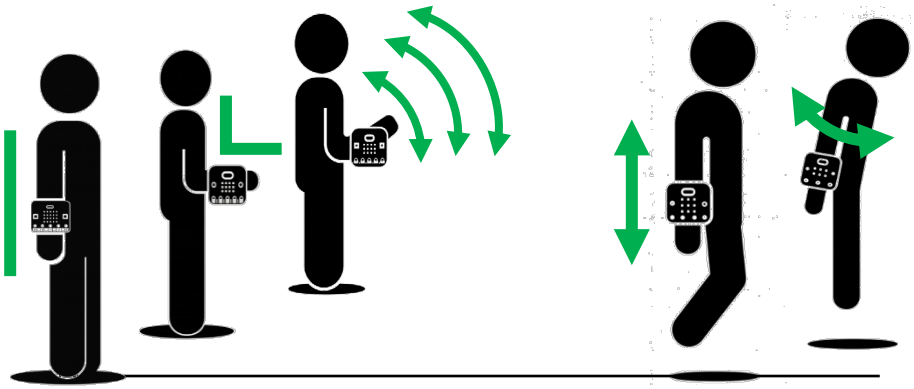
Exemple de programme : ML-microbit-Jump-Trainer-4

Exemple de programme : ML-microbit-Jump-Trainer-6



Conditions d'entraînement du modèle IA :
Le bracelet microMOUV est fixé au poignet droit avec le symbole micro:bit orienté vers l'extérieur. Plusieurs situations d'entraînement sont prises en compte pour différencier de manière fiable la détection de sauts sur place et les périodes de récupérations.

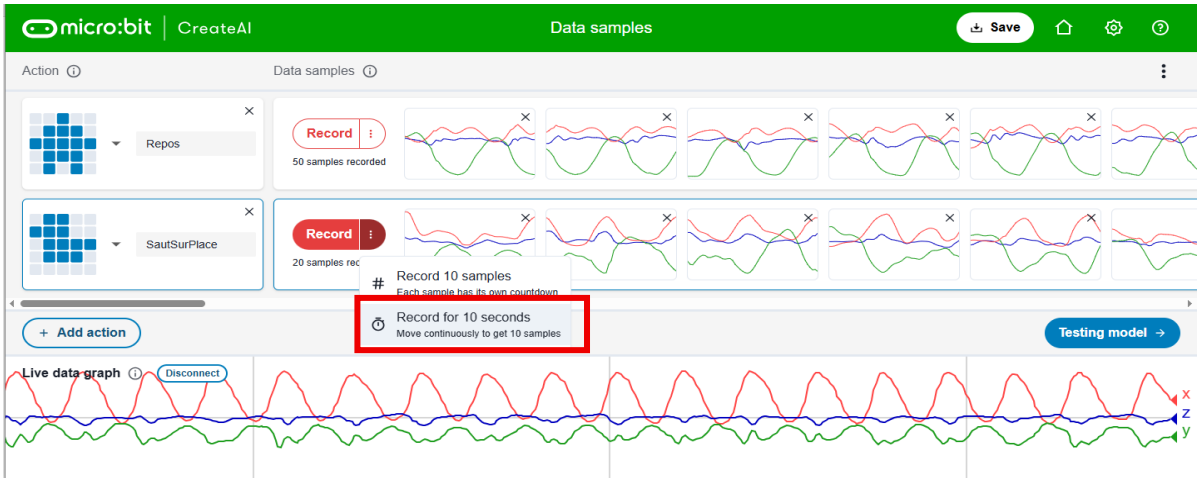
Répéter séparément les séries de gestes



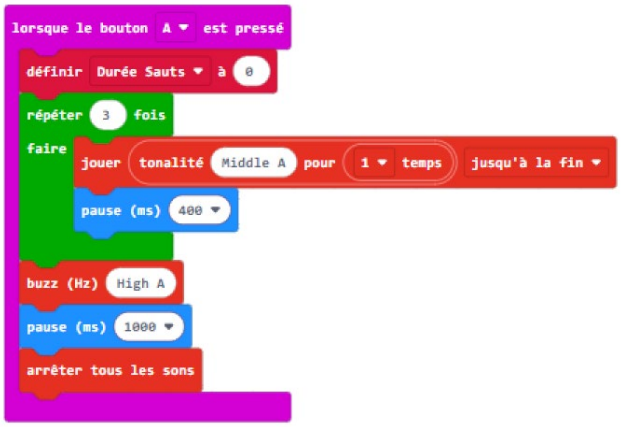
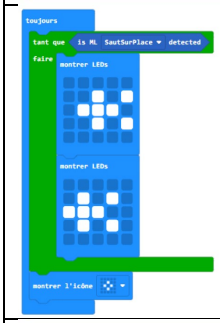
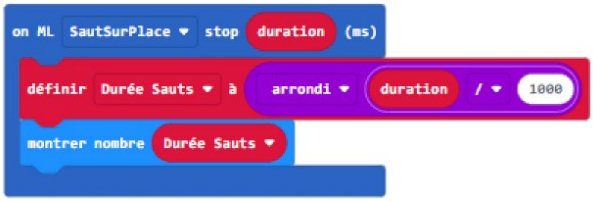
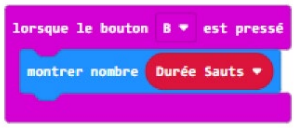
Repos
Utilisateur immobile, bras le long du corps ou avant-bras

Saut sur place
Bras le long du corps ou léger balancement durant le saut

Apprentissage :
Enregistrer séparément chaque séquence de mouvements répétés en utilisant le mode d'enregistrement pendant 10 secondes. Vérifier le modèle, enrichissez-le en associant des séquences de gestes aux types de mouvements que l'IA devra reconnaître. Vérifier la fiabilité de détection des 2 types de mouvements (repos ou Saut sur place). Ajuster éventuellement les seuils de déclenchement pour fiabiliser la différenciation des 2 mouvements.



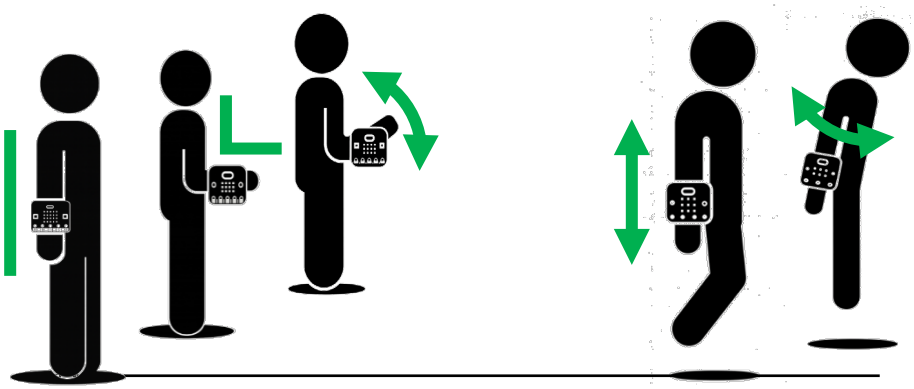
Exemple de programme : *ML_ChronoJump_V1.hex*

Bloc de programmation	Commentaire
	L'appui sur le bouton A permet d'initialiser le programme en remettant à la variable « Durée Sauts » et de déclencher une séquence sonore pour indiquer à l'utilisateur le moment où il peut commencer les sauts sur place
	Animation lumineuse d'un personnage qui saute lorsque l'IA détecte qu'une séquence de sauts est en cours
	La variable « duration » consigne le temps écoulé entre le début et la fin d'une séquence de sauts. La variable « Durée sauts » est mise à jour dès qu'une séquence de sauts prend fin. Le calcul effectué permet de convertir les ms en secondes et d'arrondir le résultat.
	Le bouton B permet de lancer l'affichage du temps consacré à la dernière série de sauts.

Cardio jump

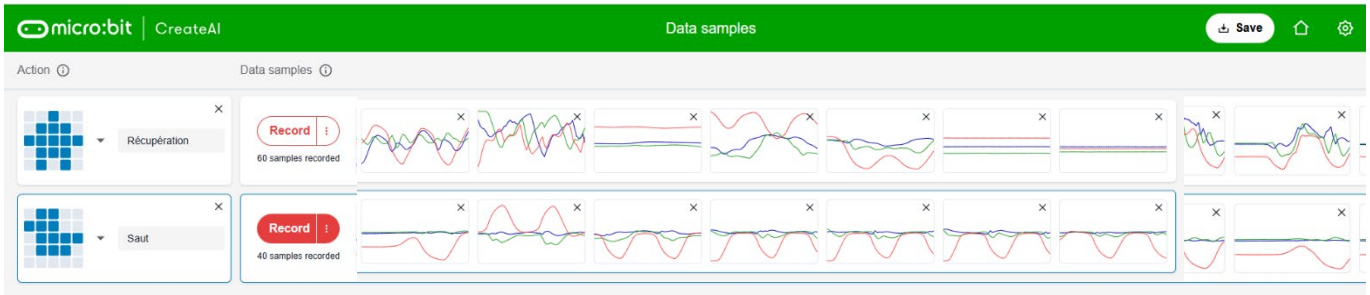
But du programme :
Programmer le bracelet microMOUV afin de réaliser un système d'entraînement pour améliorer son endurance lors de séries répétées de sauts sur place. Le bracelet est fixé au poignet ; il émet des bips différenciés toutes les 5 secondes pour indiquer alternativement le début d'une séquence de sauts puis de récupération. Le nombre cumulé de sauts effectués est affiché pendant les périodes de récupération. L'appui sur le bouton A provoque l'affichage du nombre total de sauts comptabilisés. L'appui sur le bouton B réinitialise le bracelet pour lancer une nouvelle séquence d'entraînement.

Conditions d'entraînement du modèle IA :
Le bracelet microMOUV est fixé au poignet droit avec le symbole micro:bit orienté vers l'extérieur. Plusieurs situations d'entraînement sont prises en compte pour différencier de manière fiable la détection de sauts sur place et les périodes de récupérations.

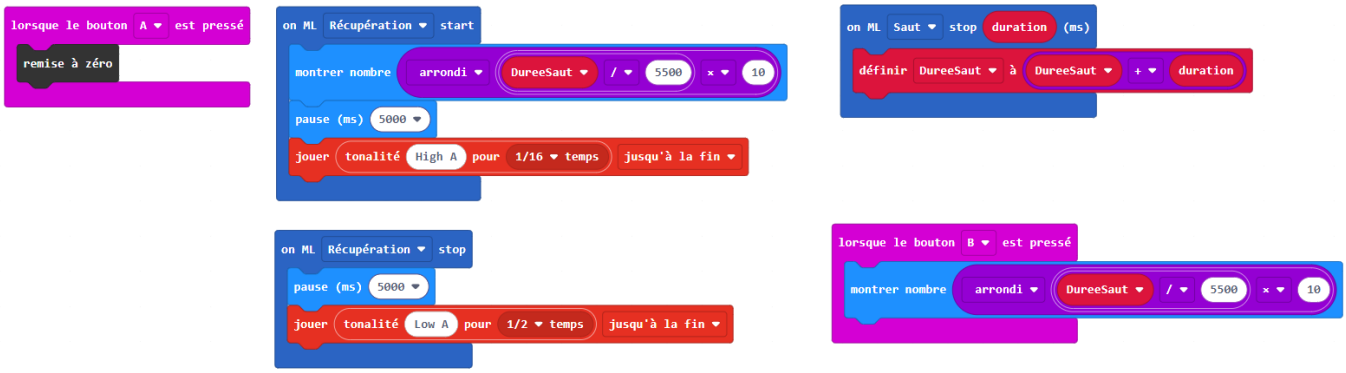


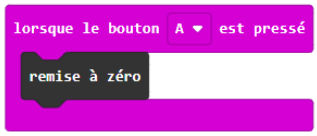
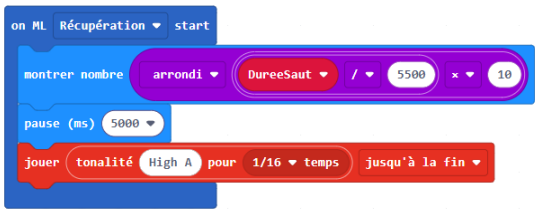
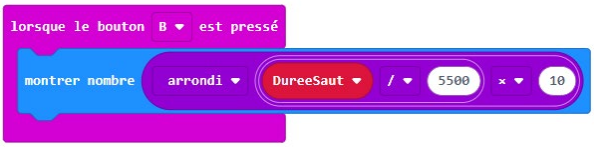
Récupération
Utilisateur immobile, bras le long du corps ou avant-bras

Saut sur place
Bras le long du corps ou léger balancement durant le saut



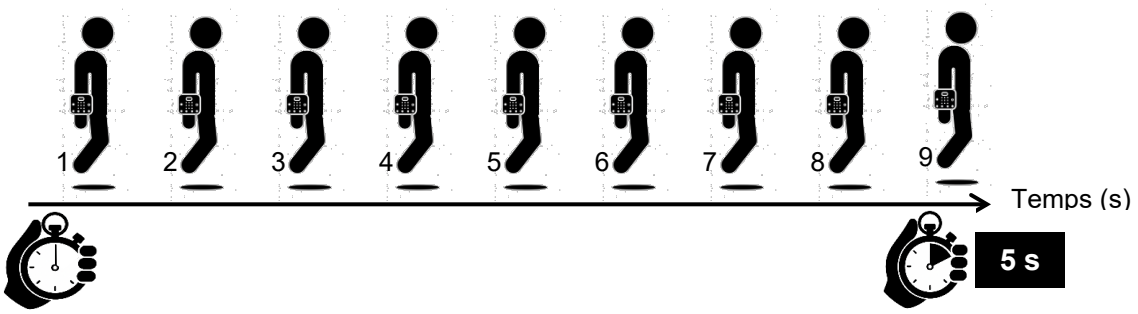
Exemple de programme : *ML_CardioJump.hex*



Bloc de programmation	Commentaire
	L'appui sur le bouton A permet d'initialiser le programme en remettant à 0 toutes les variables pour lancer la séquence d'entraînement.
	Au lancement de la séquence d'entraînement, l'utilisateur est au repos (immobile) ; il et attend le bip de départ qui survient après 5s. Le nombre affiché correspond au nombre de sauts effectués pendant la période de sauts. Le calcul effectué et le choix du coefficient « 5500 » x « 10 » sont détaillés plus loin (*)
	
	La variable DureeSaut mémorise le temps pendant lequel le saut à eu lieu (durée en ms)
	

(*) Calcule du nombre de sauts réalisés pendant une période donnée :

On considère que l'utilisateur saute de manière continue à une fréquence régulière pendant son entraînement. Il s'agit de transposer le temps total pendant lequel il saute en un nombre de sauts. Pour cela on relève le nombre réel de sauts effectués pendant 5 secondes.
À l'aide d'un chronomètre déclenché manuellement, relever le nombre moyen de sauts successifs effectués à un rythme régulier pendant des périodes de 5s.



Dans cet exemple, 9 sauts sont réalisés en 5,0 secondes, soit 9 / 5 sauts chaque seconde.
Le nombre total de sauts réalisés pendant une durée donnée correspond au nombre de sauts réalisés chaque seconde 1s et multiplié par la durée totale pendant laquelle l'utilisateur saute, soit :
Nombre de sauts = 10 /
Cela revient à dire que pendant une durée de 5,5 secondes 10 sauts sont effectués.
En considérant que l'on sautera toujours au mêm rythme, on peut
Pendant ce temps, la durée de la période de saut est collectée dans la variable « duration ». La variable « DuréeSaut » est mise à jour avec le temps pendant lequel l'utilisateur a sauté. Lorsque le bip rave et long indiquant la période de récupération survient,

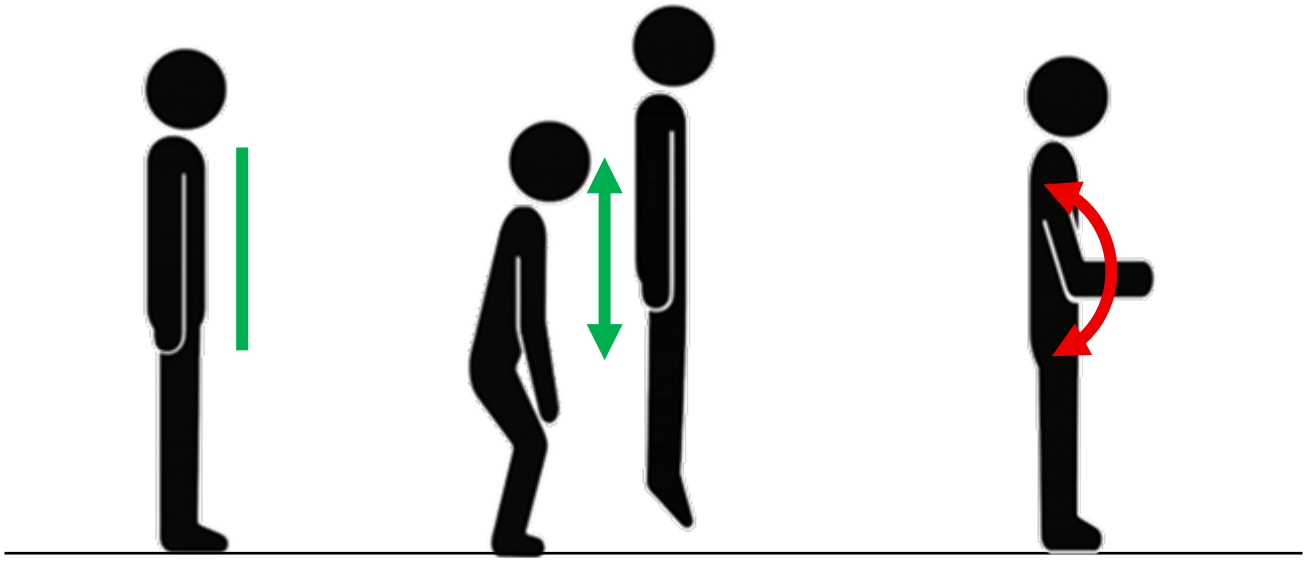
Saut / Récupération

Contexte :
Créer un système d'entraînement sportif destiné à rythmer une série de sauts sur place alternés avec des temps de récupération.

Matériel nécessaire : microMOUV

Description du programme :
Un son est émis pour indiquer à l'utilisateur d'effectuer une série continue de sauts sur place en maintenant les bras le long du corps. Un deuxième son est émis après 10 secondes pour indiquer le début de la période de récupération de 10 secondes. Cette séquence est répétée 3 fois.
À l'issue de la séquence, le nombre de sauts effectués est affiché sur la matrice de LED.

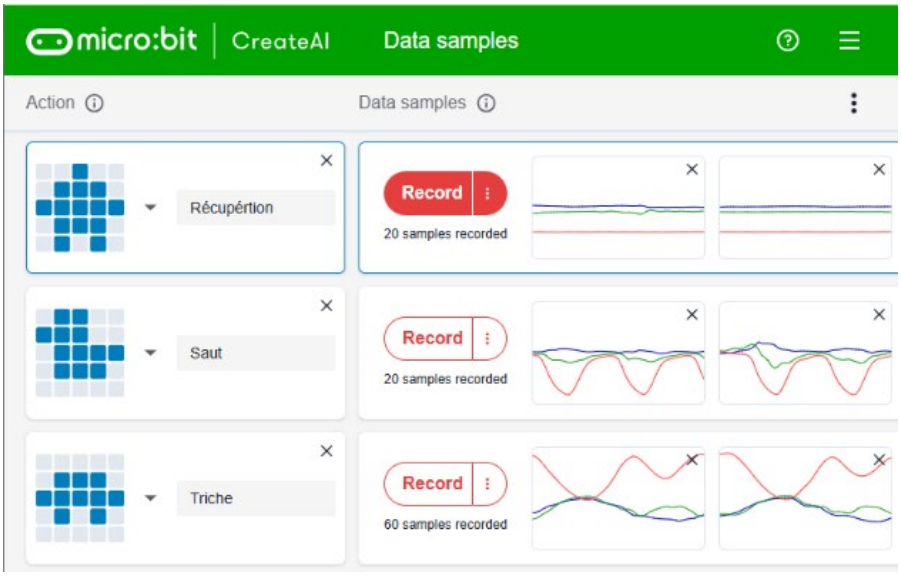
Modèle d'apprentissage : ***ML-Sauts-Récup-Triche.hex***
Ce fichier est contient le modèle d'apprentissage de la carte micro:bit pour détecter le saut sur place, la position de récupération et les mouvements de « triche » qui pourraient être assimilés au saut sur place. L'entraînement du modèle est réalisé en portant le bracelet microMOUV au poignet gauche ou au poignet droit.



Récupération
Bras le long du corps

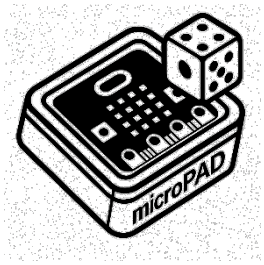
Saut
Bras le long du corps

Triche
Mouvement des avants bras sans sauter



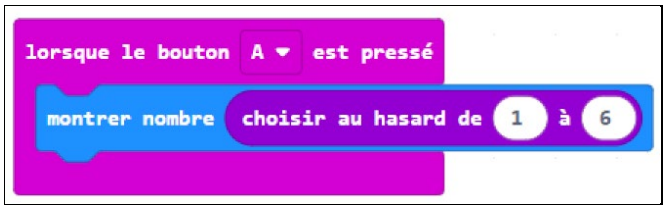
Activités de programmation avec microPAD

Dé électronique



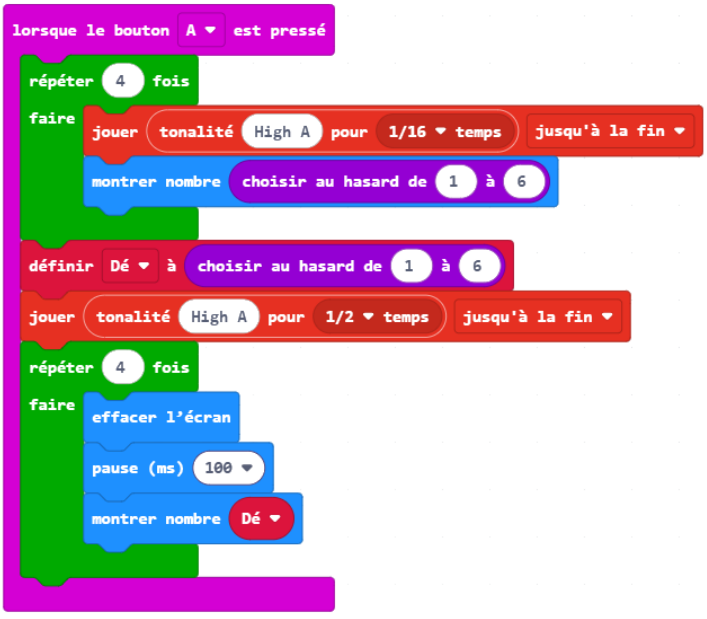
But du programme : afficher un chiffre au hasard entre 1 et 6 après l'appui sur le bouton A

Exemple de programme : ***microbit-Dé-électronique-Simple.hex***



Exemple de programme : ***microbit-Dé-électronique-Animé.hex***

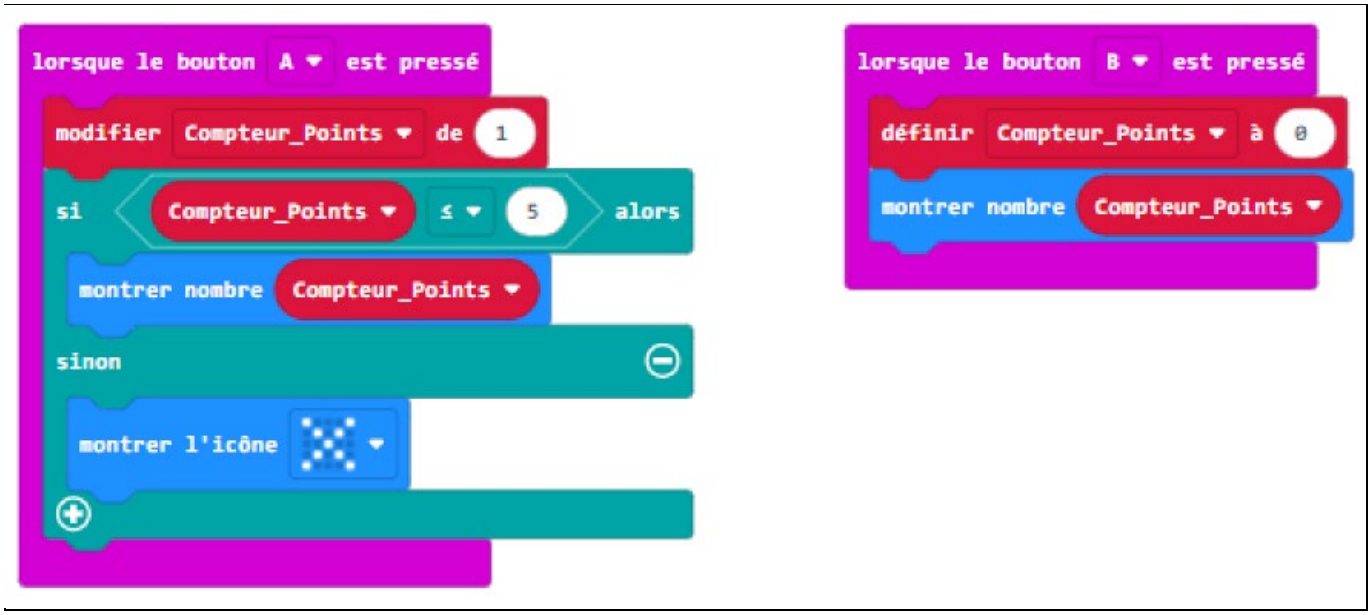
Exemple de programme : ***microbit-Dé-électronique-Animé-Cligno.hex***

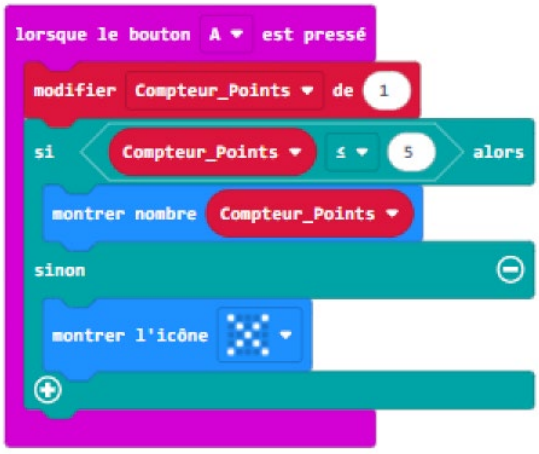
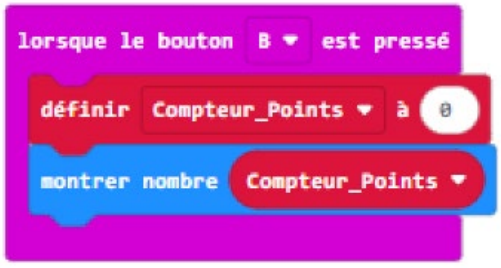


Compteur de points

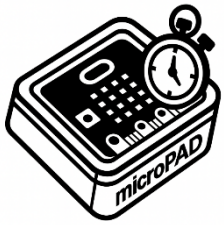
But du programme : comptabiliser un nombre de points marqués jusqu'à 5 points. L'appui sur le bouton A incrémente de 1 en 1 un compteur de points qui s'affiche sur la matrice de LED. L'appui sur le bouton B remet le compteur à 0.

Exemple de programme : ***microbit-Compteur_Points.hex***



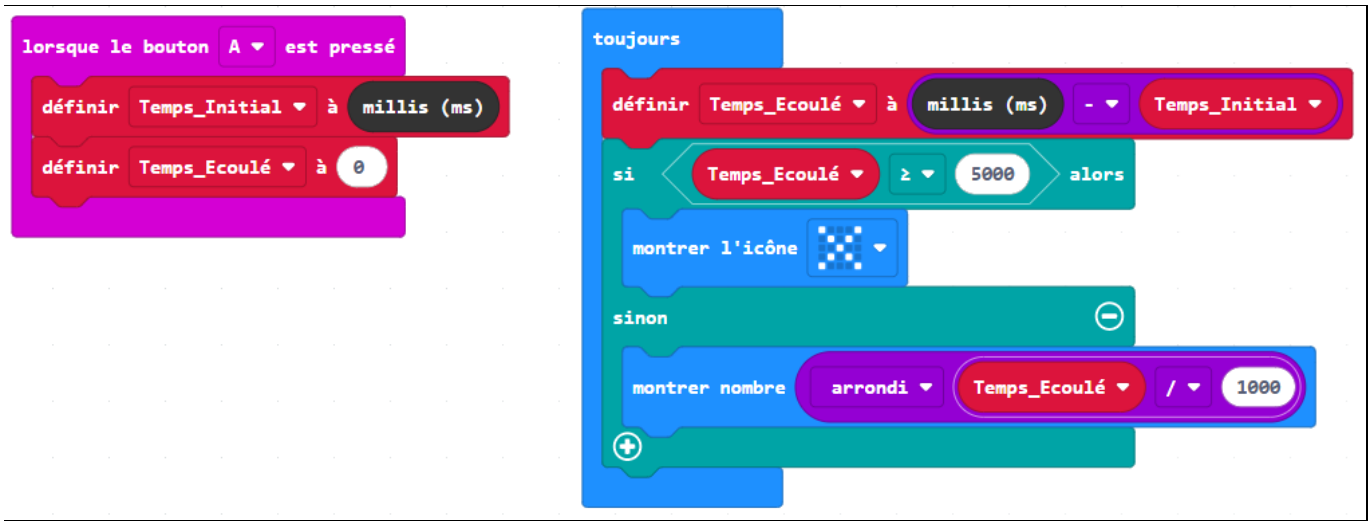
Bloc de programmation	Commentaire
	L'appui sur le bouton A incrémente de 1 en 1 la variable « Compteur_Points » et l'affiche sur la matrice de LED. Si le tirage est supérieur à 5 le symbole « X » s'affiche.
	L'appui sur le bouton B remet à 0 la variable « Compteur_Points » et l'affiche sur la matrice de LED.

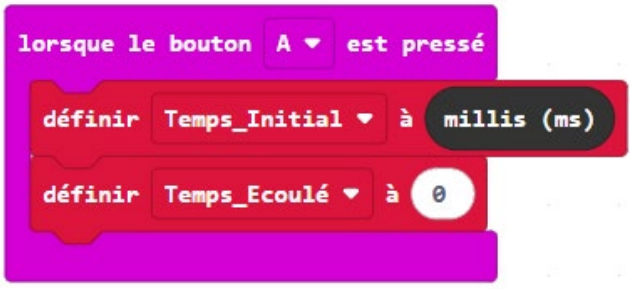
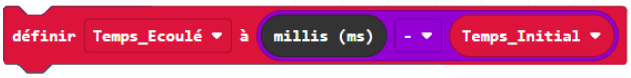
Timer 5 secondes



But du programme : déclencher et afficher toutes les secondes un timer de 5s lors de l'appui sur le bouton A. Lorsque le temps de 5 secondes est atteint, afficher le symbole « X »

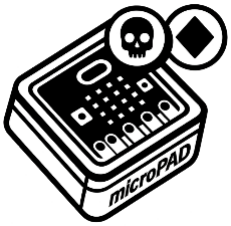
Exemple de programme : *microbit-Timer-5s.hex*



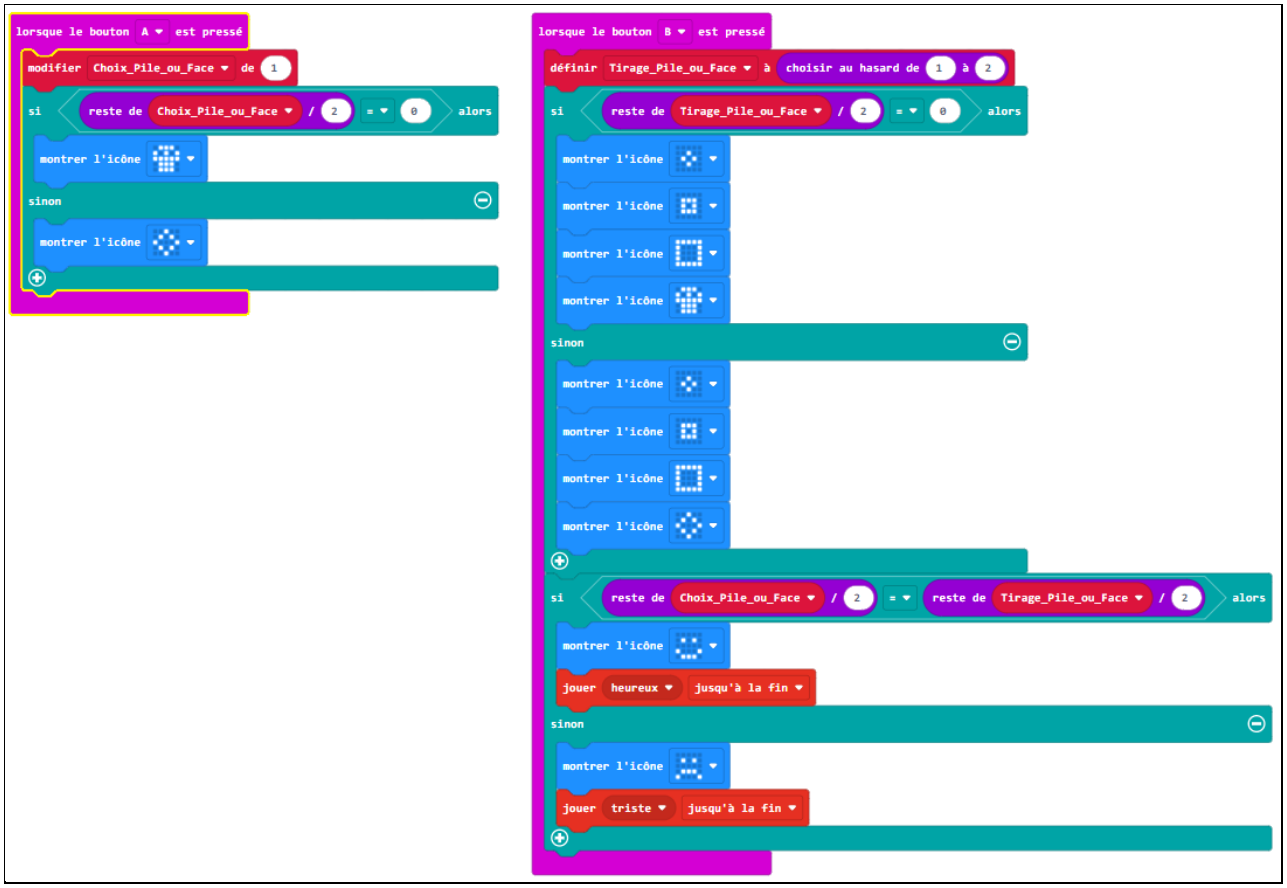
Bloc de programmation	Commentaire
	La carte micro:bit dispose d'un chronomètre interne qui compte le nombre de millisecondes écoulées depuis son démarrage. L'appui sur le bouton A initialise la variable « Temps_initial » à la valeur du chronomètre interne.
	La variable « Temps_Ecoulé » correspond à la différence entre le temps actuel du chronomètre interne et le temps initial mémorisé lors de l'appui sur le bouton A (en millisecondes)
	La partie entière du temps écoulé divisé par 1000 correspond au temps écoulé exprimé en secondes

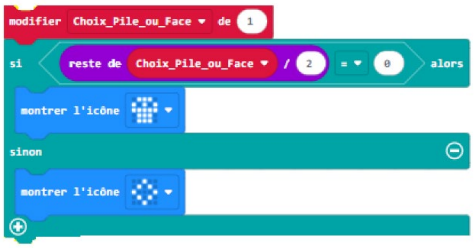
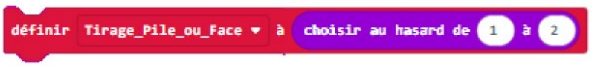
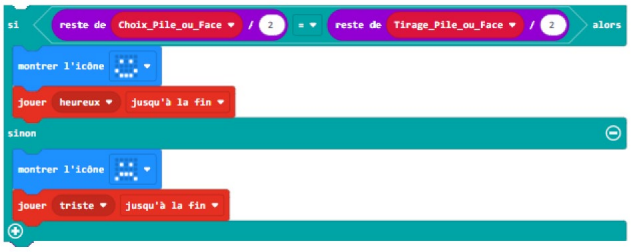
Pile ou Face

But du programme : le joueur sélectionne un symbole Pile ou Face avec le bouton A de la carte. Il lance un tirage au hasard de Pile ou Face avec le bouton B.
Après une animation visuelle, le résultat du tirage s'affiche sur la matrice de LED. Si le choix fait par l'utilisateur correspond au tirage, alors un smiley souriant s'affiche accompagné d'un son joyeux, sinon un smiley triste s'affiche accompagné d'un son triste.



Exemple de programme : ***microbit-Pile_Ou_Face.hex***

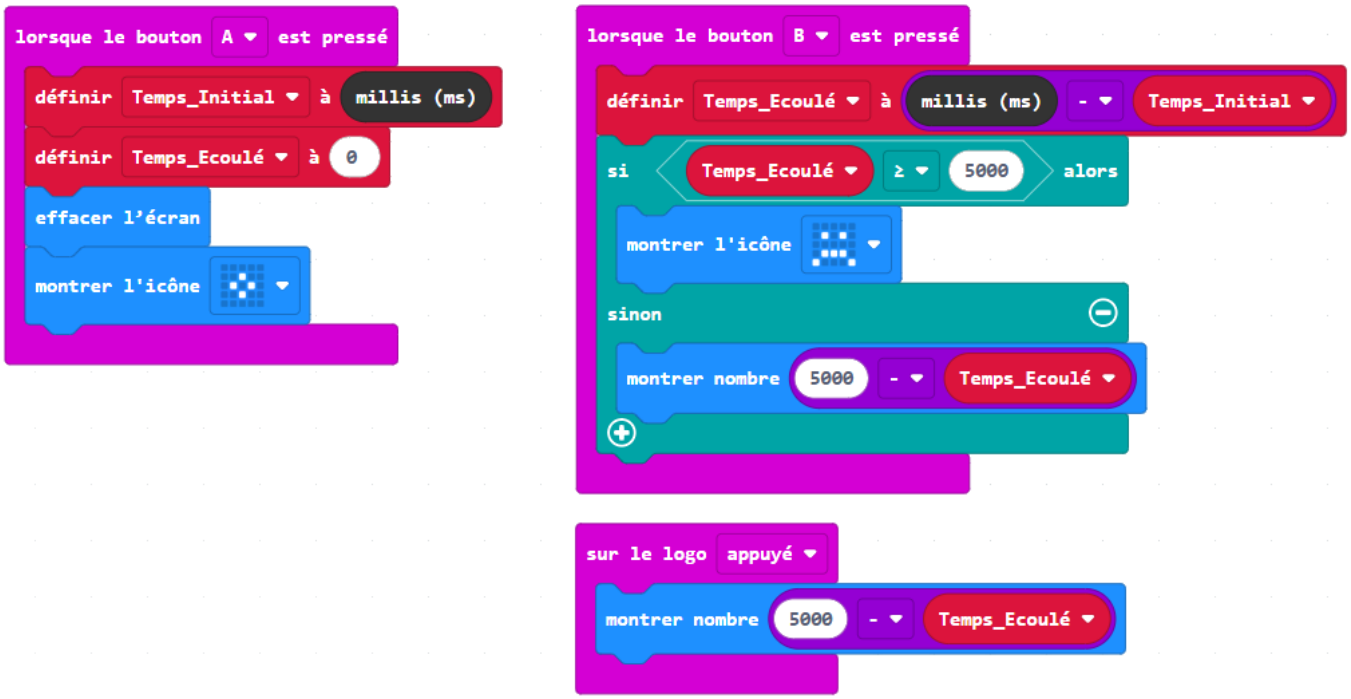


Bloc de programmation	Commentaire
	L'appui sur le bouton A incrémente de 1 en 1 la variable « Choix_Pile_ou_Face puis affiche le symbole Face si elle est paire ou Pile si elle est impaire
	L'appui sur le bouton B déclenche le tirage au hasard d'un nombre pair (2) ou impair (1)
	Le choix pair ou impair fait par l'utilisateur est comparé au tirage au hasard pair ou impair. S'ils sont équivalents, alors le joueur gagne son pari sinon il le perd.

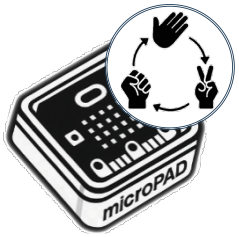
5 secondes chrono

But du programme : le joueur lance un chronomètre avec le bouton A de la carte. Il doit appuyer sur le bouton B avant que le chronomètre atteigne un temps de 5s. Si le temps est dépassé, le jeu est perdu, un smiley triste apparaît. Si le bouton B est appuyé avant que le temps de 5s ne soit écoulé, alors la différence de temps exprimée en millisecondes avec le temps de 5s s'affiche sur la matrice LED. L'appui sur le Logo microbit rappelle l'affichage de la différence de temps calculée.

Exemple de programme : ***microbit-5-Secondes-Chrono.hex***

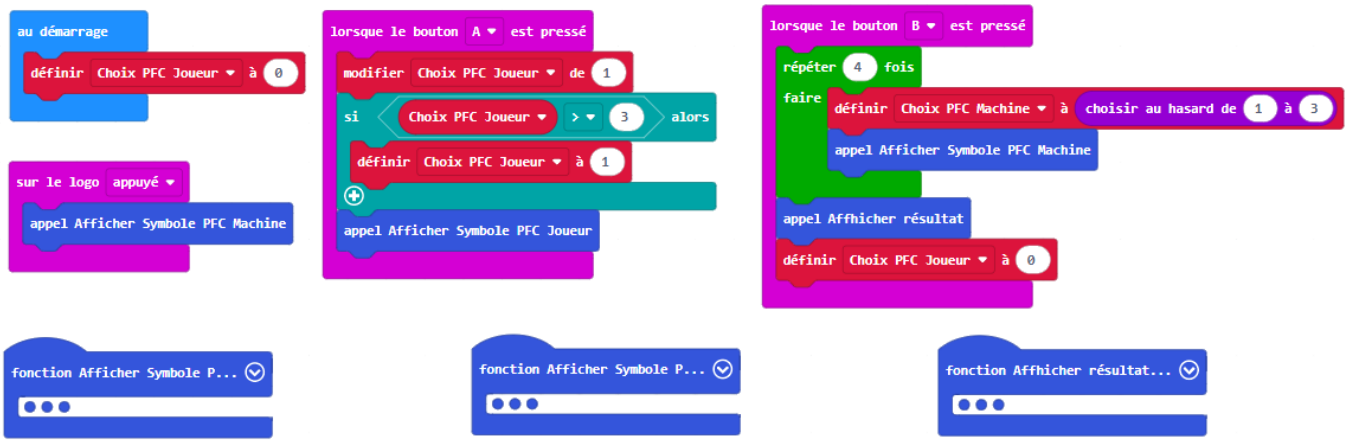


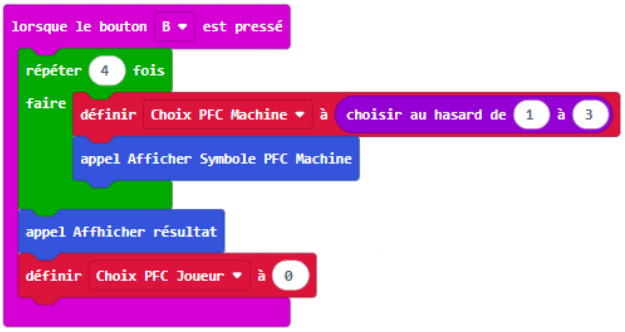
Pierre / Feuille / Ciseaux VS machine



But du programme : le joueur sélectionne Pierre (P), Feuille (F), ou Ciseau (C) avec le bouton A puis lance le jeu avec le bouton B. Le microPAD choisit alors au hasard un des 3 objets (P, F,C) puis l'affiche. Une animation précède l'affichage du vainqueur en affichant « 0 » s'il y a égalité, « J » si le joueur gagne, « M » si la machine gagne.

Exemple de programme : ***microbit-PFC-affichage-Gagnant.hex***



Bloc de programmation	Commentaire
	L'appui sur le bouton A incrémente « en boucle » la variable « Choix PFC Joueur » de 1 à 3 puis appelle la sous fonction « Afficher Symbole PFC ». Celle-ci affiche les lettres P, F ou C en fonction de la valeur de la variable.
	L'appui sur le bouton B initialise la variable « Choix PFC Machine » avec une valeur comprise entre 1 et 3. La fonction « Afficher Symbole PFC Machine » est appelée à 4 reprises pour ménager un suspens avant l'affichage définitif du symbole associé. Enfin, la fonction « Afficher résultat » est appelée pour déterminer qui du joueur ou de la machine à gagné.
	La fonction « Afficher résultat » compare le choix fait par le joueur à celui fait au hasard par la machine. La table de vérité ci-contre indique le résultat qui doit être affiché : « 0 » si le choix fait par la machine est le même que celui fait par le joueur, « J » si le joueur gagne, « M » si la machine gagne.

		CHOIX JOUEUR		
		P (1)	**F (2)**	**C (3)**
CHOIX MACHINE	**P (1)**	0	J	M
	F (2)	M	0	J
	C (3)	J	M	0

Mémo flash

microPAD maitre microPAD jeu

Description du jeu : le jeu consiste à observer des motifs qui s'affichent de manière aléatoire et brève sur des microPAD et à sélectionner le (les) microPAD sur lequel (lesquels) est apparaît un motif cible proposé au début du jeu. La capacité d'observation du joueur peut être mise à l'épreuve en proposant des motifs qui présentent des différences minimes.

Exemple de programme : *microbit-MemoFlash.hex*
Décomposition du programme :

1) Phase d'initialisation

microPAD maître
- Initialisation de la communication radio
- Constitution de la liste des motifs
- Envoi du motif cible Détermination du motif cible choisi au hasard par le microPAD maître
- Envoi par radio du numéro motif cible vers les microPAD de jeu

microPAD jeu
- Initialisation de la communication radio
- Affichage du motif cible

2) Phase de lancement
microPAD maître
- Constitution de la liste des microPAD de jeu
- Détermination du motif cible choisi au hasard par le microPAD maître
- Envoi par radio du motif cible vers les microPAD de jeu

microPAD jeu
- Initialisation de la communication radio
- Affichage du motif cible
- Déclenchement par (appui simultané sur les boutons A+B du microPAD maître
- Définition de la liste des motifs
- Affichage du motif cible pendant 1 seconde sur tous les microPAD de jeu

3) Phase de fin

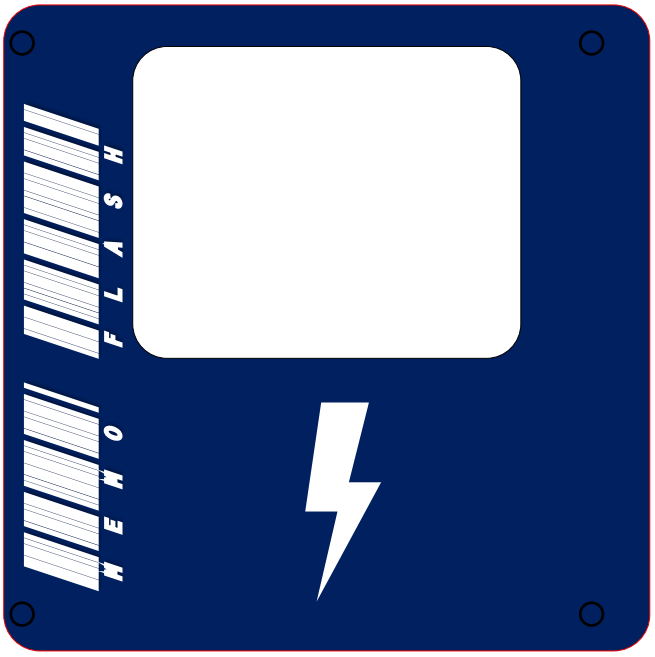
- Affichage d'un motif au hasard sur chaque microPAD
- Détermination du motif cible choisi au hasard par le microPAD maître
- Envoi par radio du motif cible vers les microPAD de jeu
- Affichage du motif cible pendant 1 seconde sur tous les microPAD de jeu

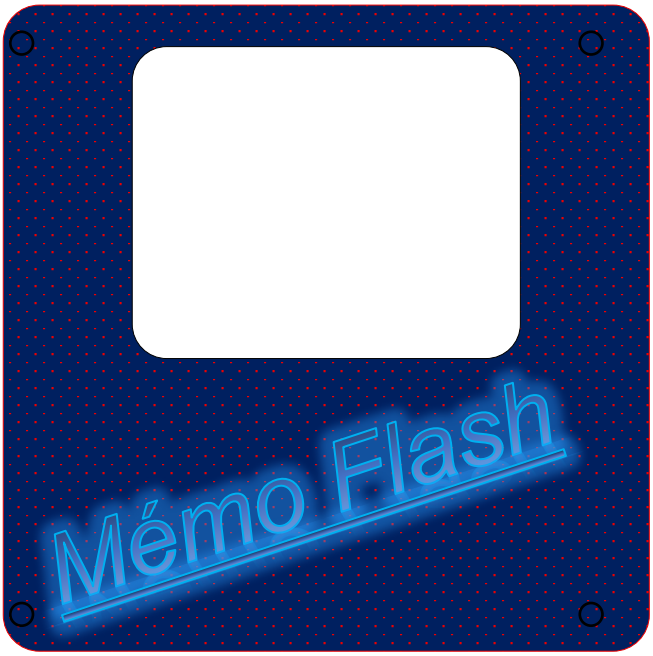
- Affichage d'un motif au hasard sur chaque microPAD
- Détermination du motif cible choisi au hasard par le microPAD maître
- Envoi par radio du motif cible vers les microPAD de jeu
- Affichage du motif cible pendant 1 seconde sur tous les microPAD de jeu

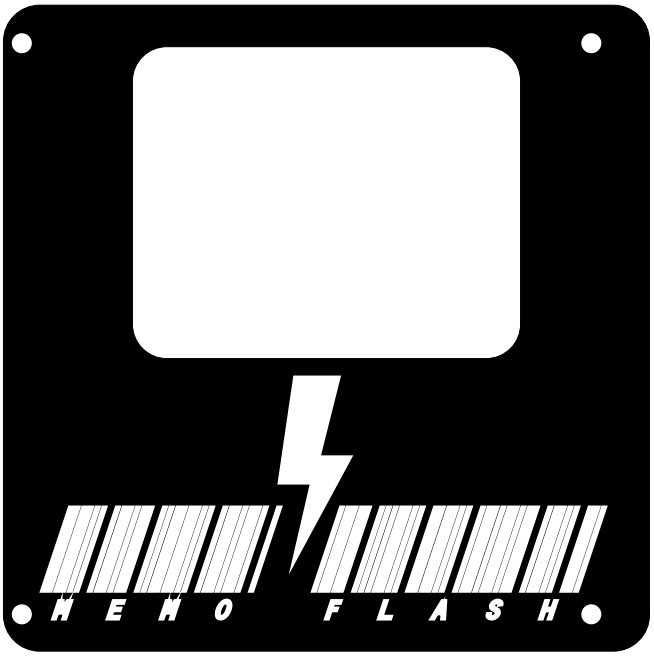
4) Phase d'affichage du résultat

- Afficher le motif cible sur tous les microPAD
- Générer des motifs au hasard et les afficher sur les microPAD
- Compter le nombre d'apparitions du motif cible
- Compter le nombre de sélections effectuées à l'apparition du motif cible
- Afficher un symbole « Gagné » si le joueur a sélectionné le motif cible autant de fois qu'il est apparu ; sinon, afficher un symbole « Perdu ».

Exemples de visuels à imprimer et découper : fichier Word *Visuel Word MémoFlash.docx*



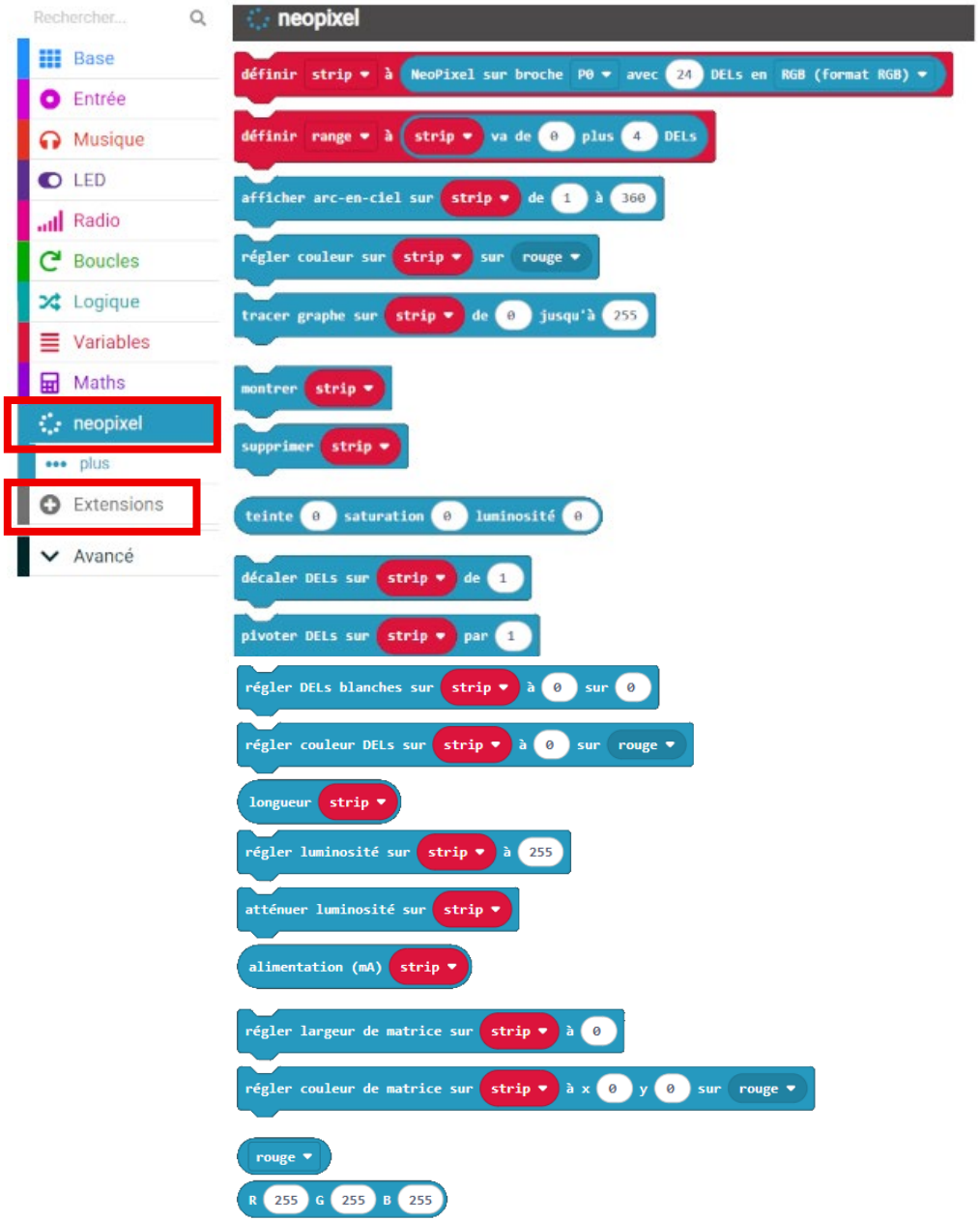




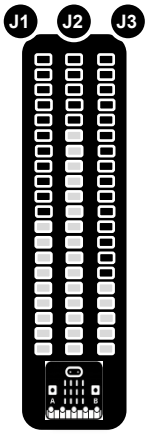
Mise en service de l'Extension neopixel

La programmation des bandes de LED nécessite d'installer l'extension Neopixel disponible dans MakeCode.

Jeu d'instructions neopixel

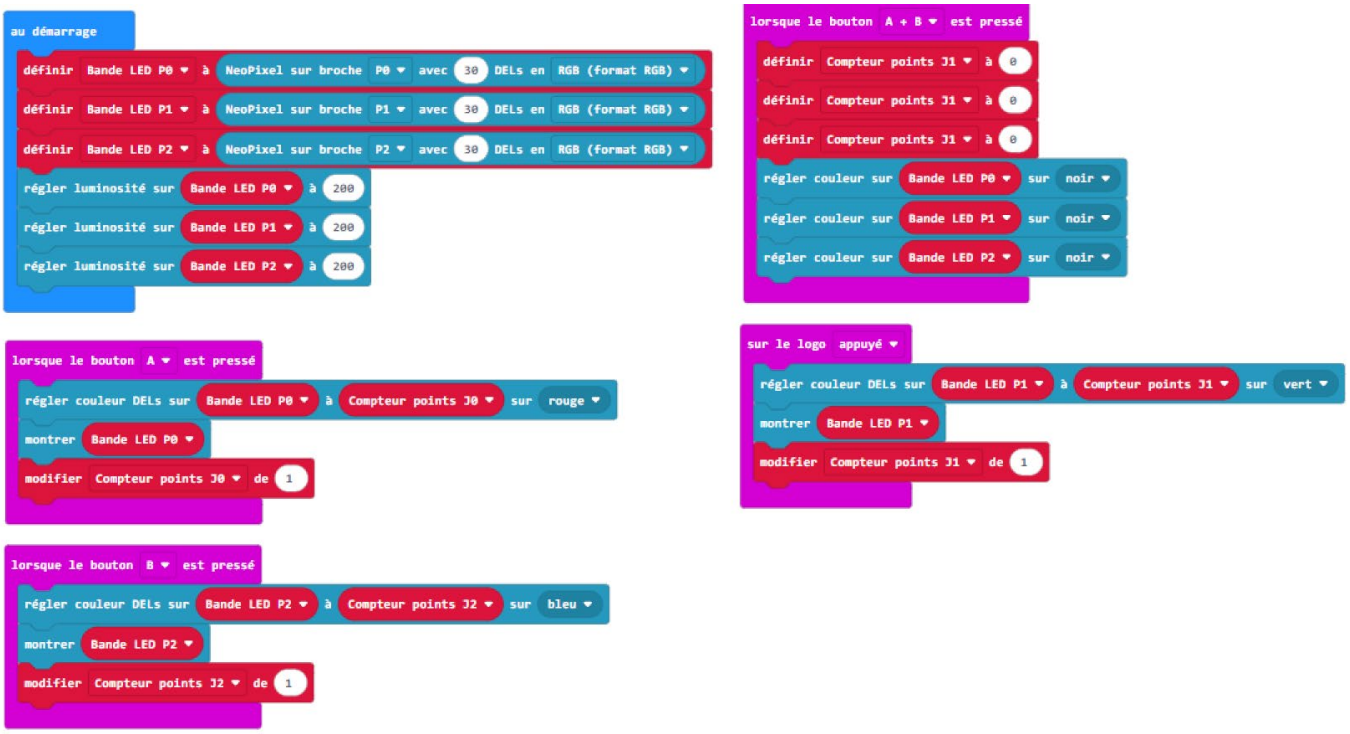


Compteur de points

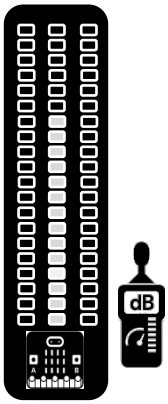


But du programme :
Programmer le panneau d'affichage microSCORE afin de comptabiliser des points de 3 joueurs différents à l'aide des 3 boutons A, B et symbole micro:bit. Chaque appui incrémente l'affichage d'une LED. Chaque bande de LED est associée à un des 3 boutons avec la première bande qui s'éclaire en rouge, la 2e en vert et la 3e en bleu.

Exemple de programme : ***microbit-Compteur-points-3-joueurs.hex***



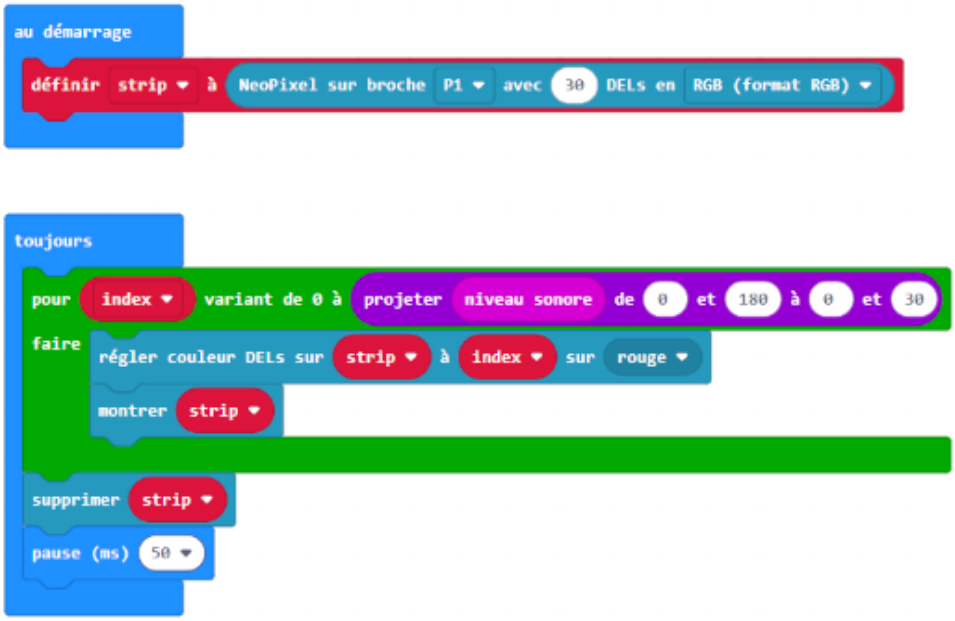
Niveau sonore

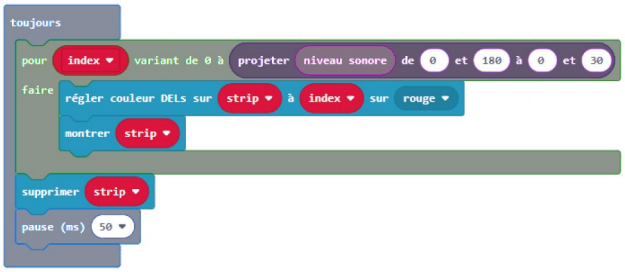


But du programme :
Programmer le panneau d'affichage microSCORE afin d'afficher sur la bande de LED centrale le niveau sonore ambiant détecté par le microphone de la carte micro:bit.

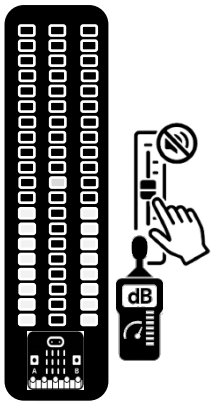
Note :
Le microphone du micro:bit mesure les niveaux sonores par des nombres compris entre 0 et 255. 0 correspond au niveau le plus faible et 255 au niveau le plus élevé qu'il puisse mesurer.

Exemple de programme : *microbit-Niveau-Sonore.hex*



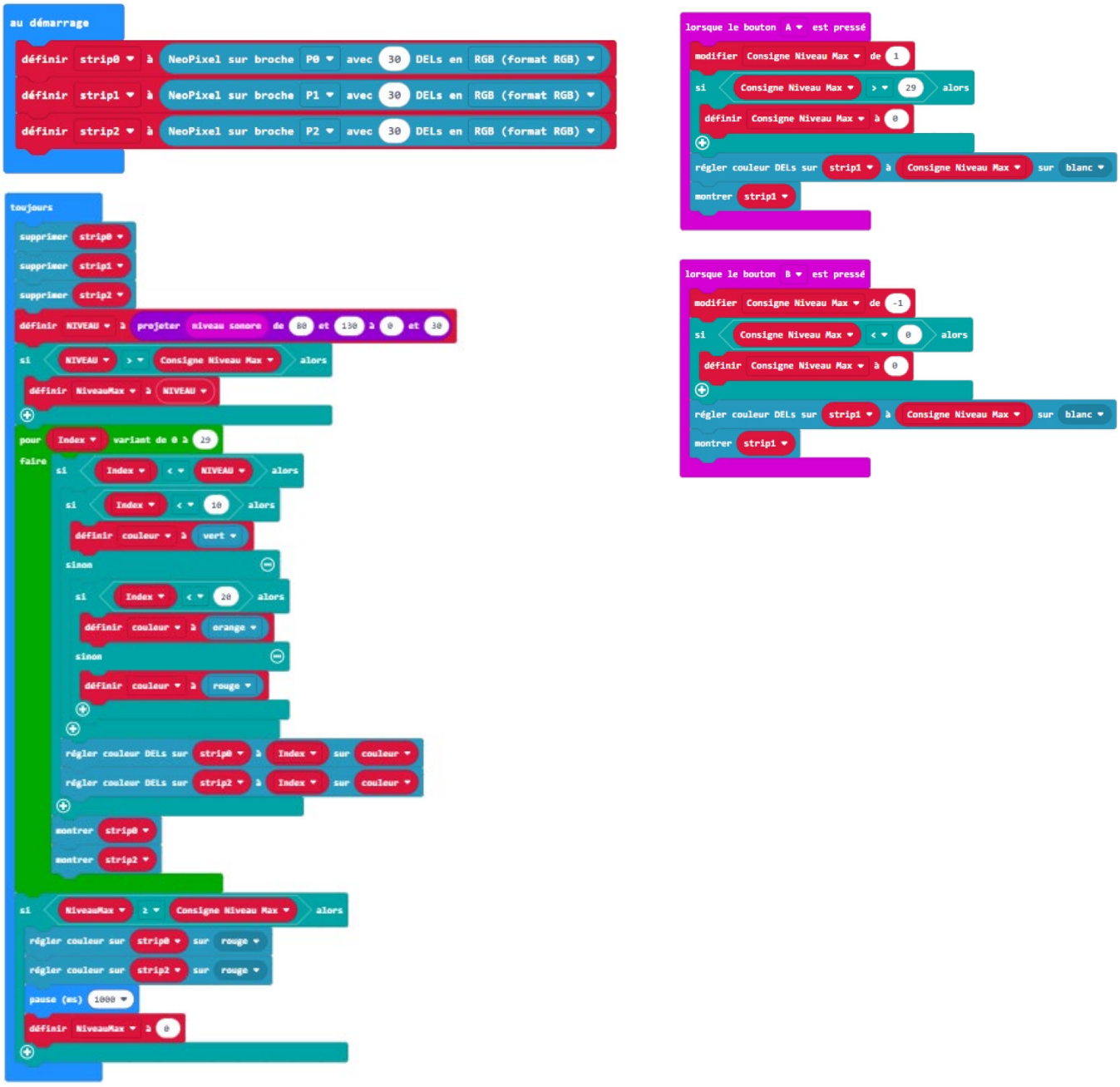
Bloc de programmation	Commentaire
	La valeur du niveau sonore captée par le microphone est convertie sur une échelle de 0 à 30. **Note :** la pleine échelle de mesure du niveau sonore est comprise entre 0 et 255. Ici elle est restreinte entre 0 et 180. Il est possible d'ajuster ces 2 valeurs pour modifier la sensibilité de détection.
	La variable index prend une valeur qui s'incrémente automatiquement de 1 en 1 entre 0 et une valeur extrême qui est fonction du niveau sonore capté par le microphone. La valeur maximum renvoyée par le microphone est 30.
	Les LED de 0 à la valeur courante du niveau sonore (30 maximum) sont allumées en rouge puis éteintes pendant 50 ms jusqu'à la prochaine itération de la boucle « pour index variant de 0 à x »

Sonomètre avancé



But du programme :
Programmer le panneau d'affichage microSCORE afin de surveiller si le niveau sonore dépasse une consigne programmée.
La consigne de niveau sonore maximum à ne pas dépasser et afficher en blanc sur la bande de LED du milieu. Les boutons A et B permettent d'incrémenter ou de décrémenter la valeur de la consigne entre 0 et 30.
Les bandes de LED gauche et droite affichent le niveau sonore en vert si le niveau sonore est inférieur à 10, en orange si le niveau sonore est supérieur à 10 et inférieur à 20, en rouge au-delà de 20.
Lorsque le niveau sonore dépasse la consigne, les bandes de LED gauche et droite s'allument en rouge pendant 1 seconde.

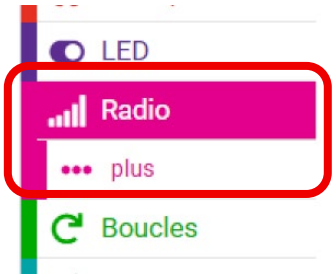
Exemple de programme : ***microbit-Sonomètre-Avancé.hex***



Exemple de programme : microbit-Sonomètre-Avancé-V2
Enrichissement du programme précédent avec une animation lumineuse au 3e dépassement de niveau sonore au-delà de la consigne programmée.

Communication radio avec microSCORE

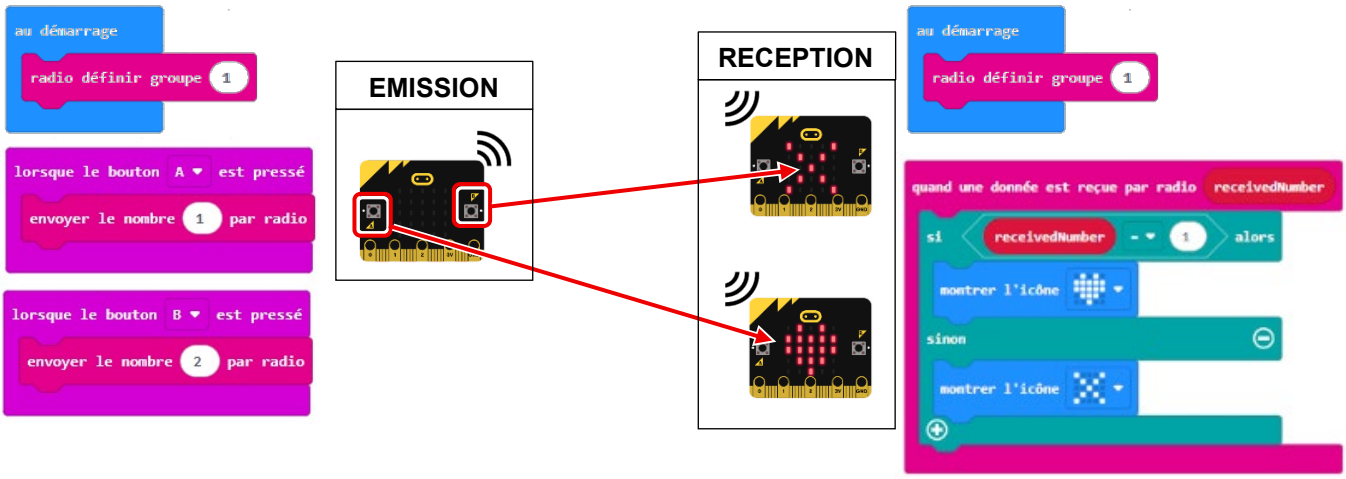
Mise en œuvre de la communication radio avec micro :bit



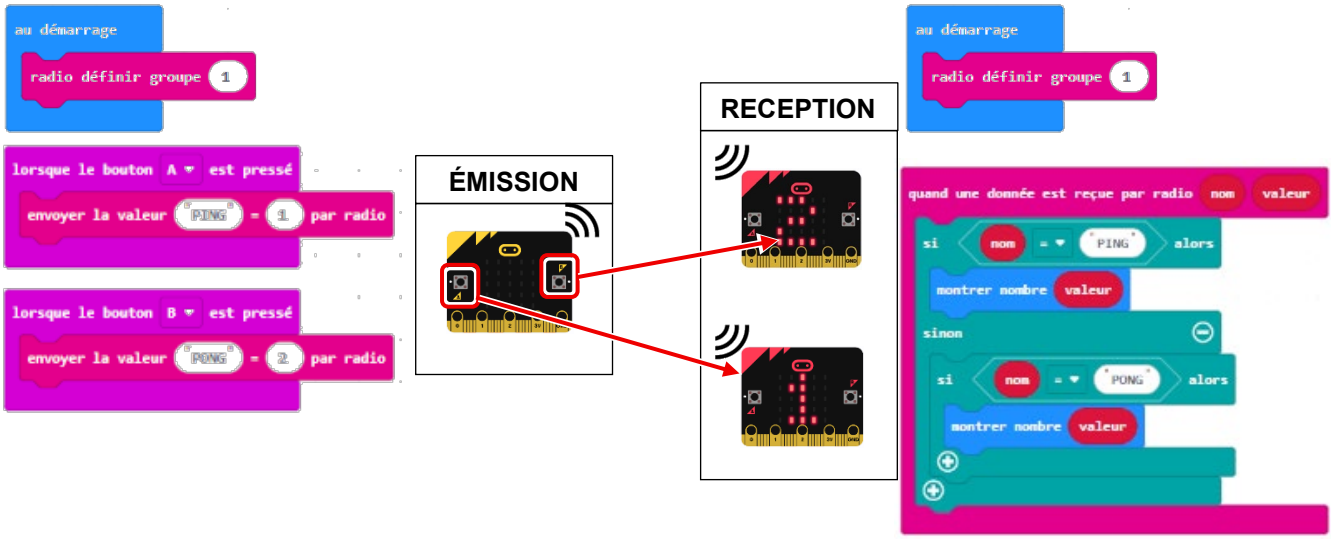
La transmission d'informations par radio offerte par la carte micro:bit permet d'établir très facilement une communication entre le bracelet microMOUV, le Pad de jeu microPAD et l'afficheur microSCORE.
Le menu « Radio » donne accès à l'ensemble des blocs utiles pour la communication sans fil entre 2 cartes micro:bit

TUTORIELS : https://makecode.microbit.org/reference/radio

Exemple de communication d'une valeur entre 2 cartes micro:bit

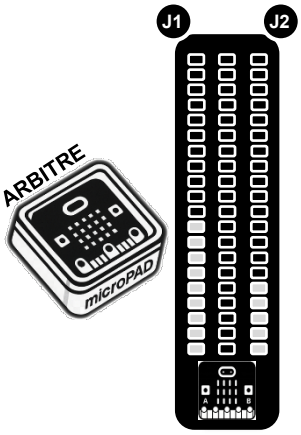


Exemple de communication d'un message véhiculant une valeur entre 2 cartes micro:bit



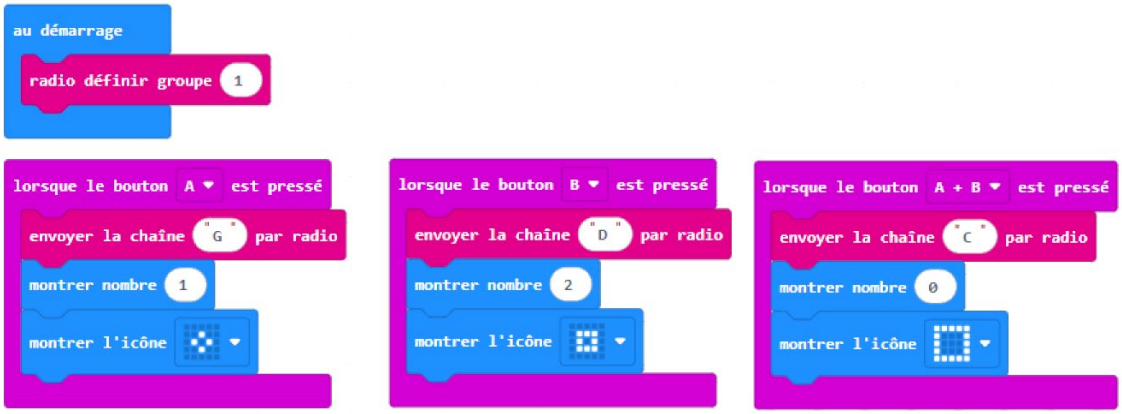
Points d'attention
Les informations communiquées par radio sont soit des textes (=chaînes de caractères=Nom= « String ») soit des nombres (= valeurs = « Number »). Les chaînes de caractères sont encadrées par des guillemets (« xxx »)
Une même carte peut émettre et recevoir (communication bidirectionnelle)
Il est possible de faire communiquer une multitude de cartes simultanément. Au-delà de 6 cartes micro:bit qui communiquent de manière asynchrone les unes avec les autres, on peut observer des ralentissements dans le traitement des communications. En théorie, une infinité de cartes peuvent communiquer les unes avec les autres, néanmoins il est souhaitable d'organiser la communication pour éviter une communication simultanée et permanente entre toutes les cartes.

Compteur de points microPAD + microSCORE



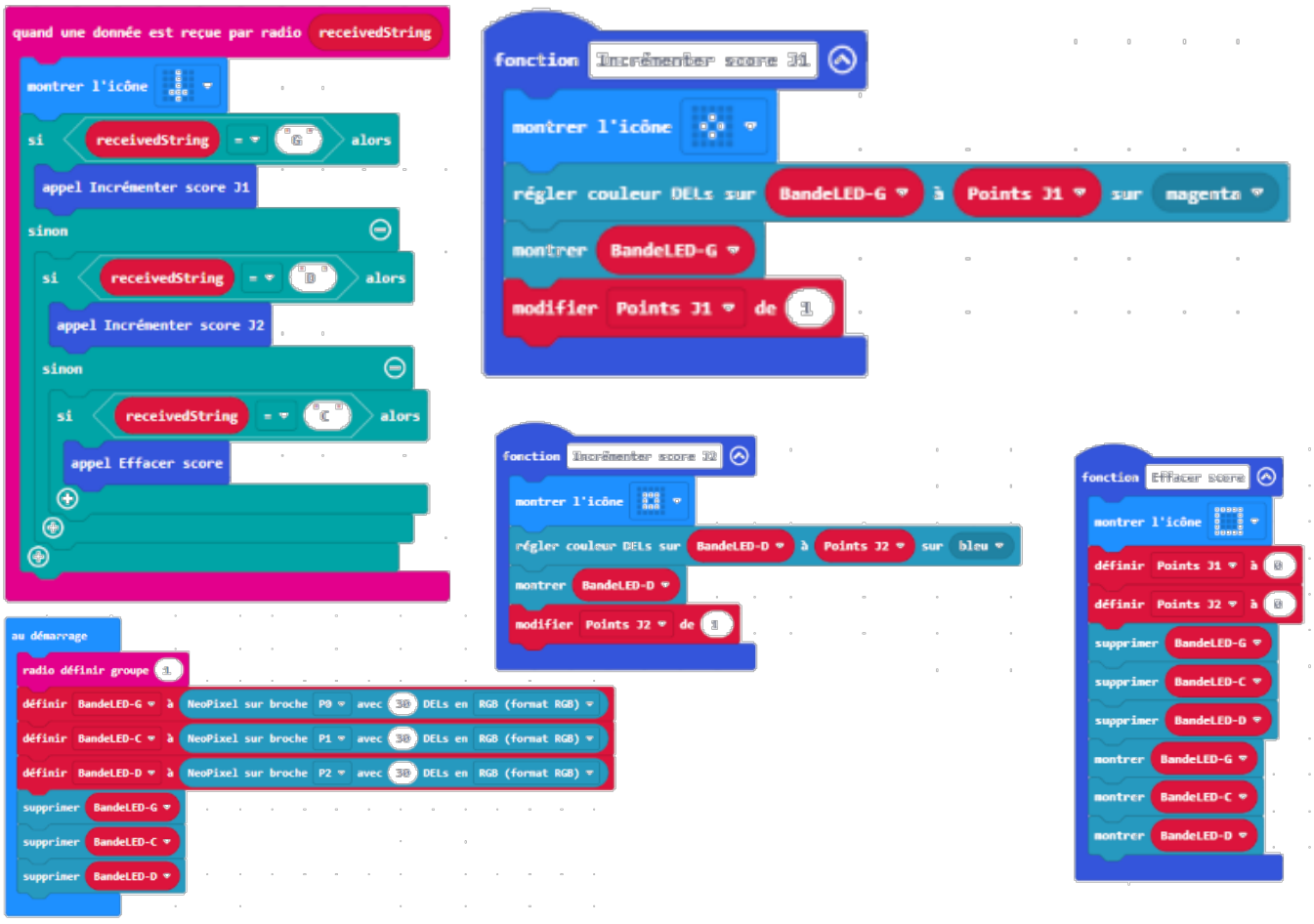
But du programme :
Programmer un microPAD pour comptabiliser les points marqués par 2 joueurs en les affichant sur microSCORE. Chaque appui sur le bouton A fait progresser d'un en un la bande de LED gauche de microSCORE (Joueur 1). Chaque appui sur le bouton B fait progresser d'un en un la bande de LED droite de microSCORE (Joueur 2). L'appui simultané sur les boutons A et B efface l'affichage des points sur microSCORE.

Exemple de programme *microPAD : microbit-MP-CompteurPoints.hex*



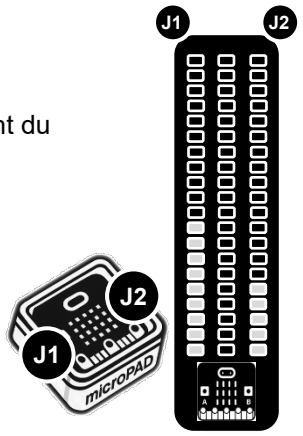
L'appui sur les différents boutons provoque l'émission par radio d'un caractère « G », « D » ou « C »

Exemple de programme *microSCORE : microbit-MS-CompteurPoints.hex*



La réception par radio d'un caractère « G », « D » ou « C » provoque l'incrément des bandes de LED concernées ou leur remise à zéro.

Horloge de jeu 2 joueurs microPAD + microSCORE



But du programme :
microPAD est utilisé comme horloge de jeu d'échecs pour 2 joueurs. Les 2 joueurs disposent du même temps de réflexion.
Lorsque le joueur A a joué, il appuie sur le bouton A pour décompter le temps du joueur B
Lorsque le joueur B a joué, il appuie sur le bouton B pour décompter le temps du joueur A

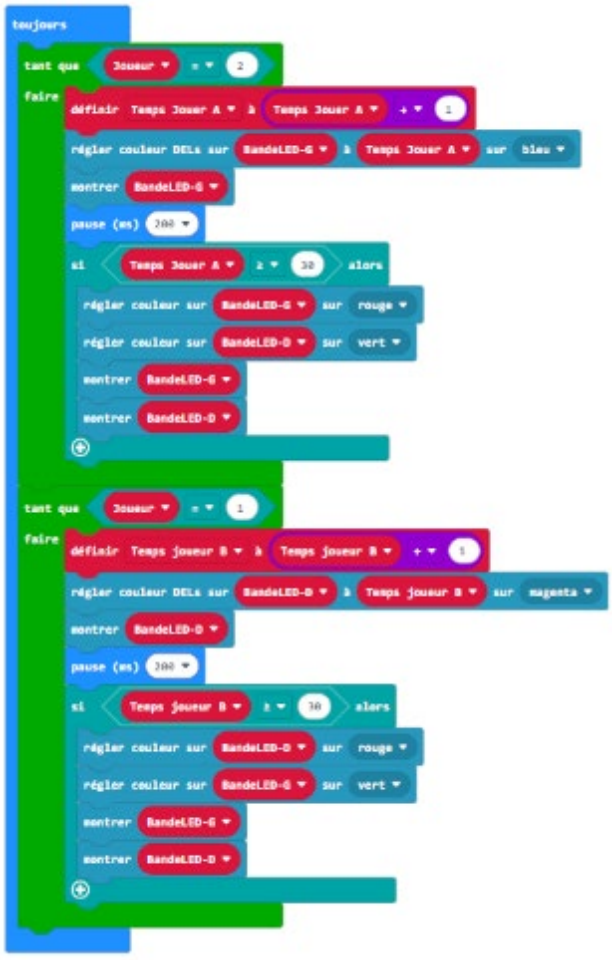
Le temps de jeu de chaque joueur est matérialisé par la progression des de la bande de LED gauche pour le joueur A et droite pour le joueur B sur microSCORE.
Si le temps imparti est écoulé pour un joueur, la bande de LED concernée clignote en ROUGE.
L'appui simultané sur les boutons A et B efface l'affichage des points sur microSCORE et remet le comptage de temps à zéro jusqu'à ce qu'un joueur appuie sur son bouton (A ou B).

Dans l'exemple ci-dessous le temps total de jeu est fixé à 200 ms x 30 = 6 secondes pour chaque joueur

Exemple de programme microPAD : *microbit-MP-Horloge 2 joueurs.hex*

L'appui sur les différents boutons provoque l'envoi d'un caractère « G », « D » ou « C »

Exemple de programme microSCORE : *microbit-MS-Horloge 2 joueurs.hex*

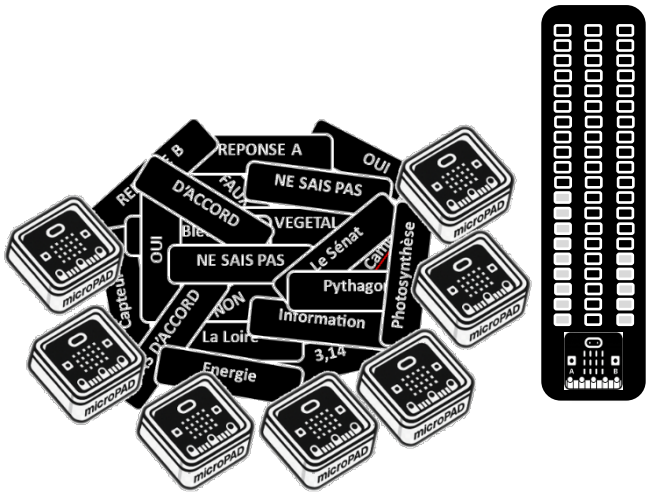


Quizz ou vote

But du programme :
Plusieurs microPAD sont utilisés comme boîtiers de vote ou de réponse à un Quizz. Le nombre de réponses s'affiche en direct sur microSCORE.
L'appui sur les boutons d'un microPAD incrémente l'affichage des bandes LED de microSCORE :

A incrémente la bande de LED gauche,
A+B incrémente la bande de LED centrale,
B incrémente la bande de LED droite

L'appui sur le bouton A de microSCORE réinitialise l'affichage en éteignant toutes les bandes LED

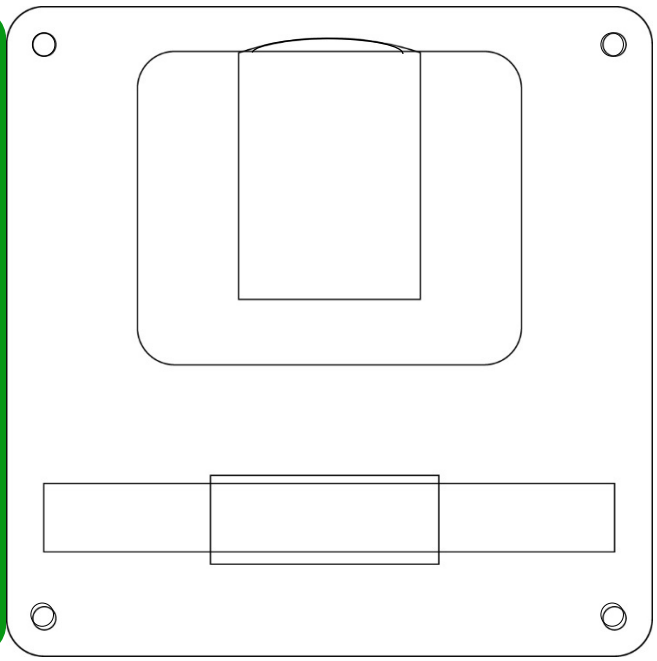


Exemple de programme microPAD : *microbit-MP-Quizz.hex*

Exemple de programme microSCORE : *microbit-MS-Quizz.hex*

ANNEXE

Exemples de visuels microPAD à imprimer et découper



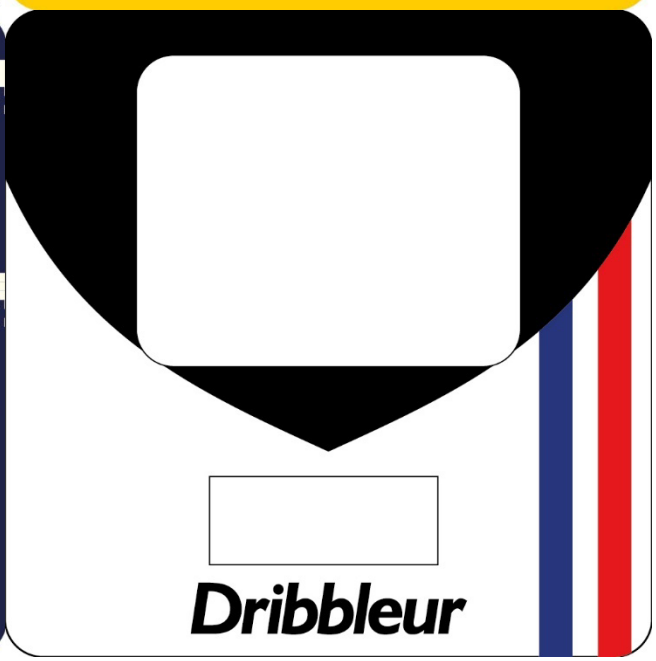

BATTLE


D'ACCORD
PAS D'ACCORD
A voté !

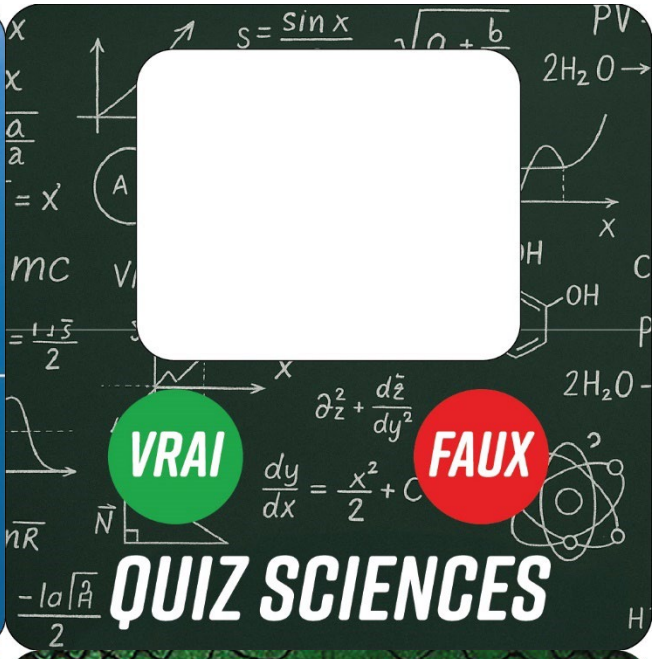

APPLAUDIMÈTRE


BUZZZ


ALIEN ATTACK


Dribbleur


PING
PONG


VRAI
FAUX
QUIZ SCIENCES


RÉFLEXES


SNAKE


SPORT GAME


TIMER

Modèles d’entraînement IA prêts à l’emploi

https://microbit.org/projects/make-it-code-it/?filters=bd4c25b7-652b-4d0d-94c8-9815d8f32cf4

Liens utiles pour aller plus loin

https://ml-machine.org/

https://microbit.org/

https://microbit.org/projects/make-it-code-it/

https://microbit.org/teach/classroom-resources/microbit-createai-glossary/

https://microbit.org/teach/do-your-bit/

www.a4.fr
Concepteur et fabricant de matériels pédagogiques