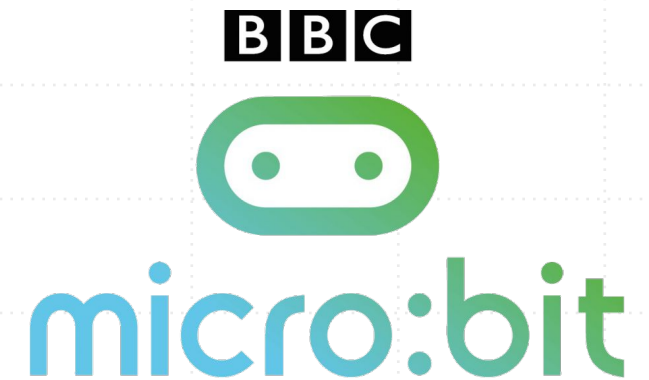




L'IOT POUR 5 ANS ET PLUS

Apprendre à coder avec micro:bit



Sommaire



- 01 ▪ La carte micro:bit

- 02 ▪ Usages

- 03 ▪ Exercices

- 04 ▪ Conclusion

- **Jonathan BARANZINI**

- Développeur



Positive innovation



- **Thomas CAMI**

- Développeur



WINAMAX

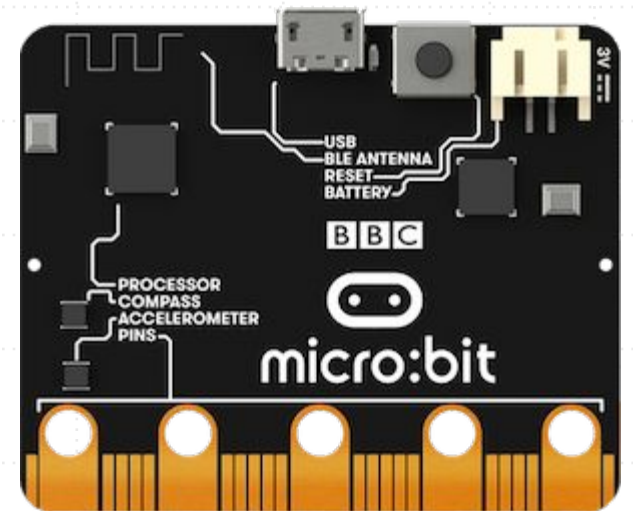
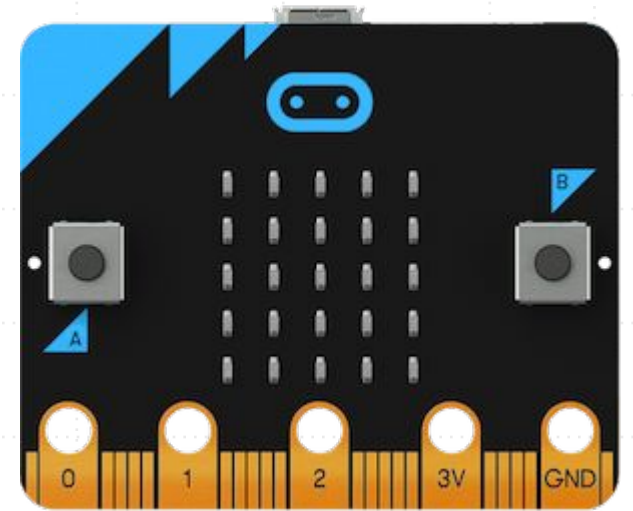




01 - La carte micro:bit

La carte micro:bit





| | |
|------------------------|---|
| Date | 06 / 07 / 2015 |
| Fabricant | BBC (British Broadcasting Corporation), Micro:bit Educational Fundation |
| Objectif | Former les jeunes au développement informatique |
| Technologie | <ul style="list-style-type: none"> • Microcontrôleur 16Mhz, 32 bits • 256ko mémoire flash + 16ko mémoire vive • Matrice led 5 x 5 |
| Capteurs | <ul style="list-style-type: none"> • 2 boutons + 1 bouton sensible • 1 thermomètre, 1 boussole • 1 capteur de mouvement 3D • 1 micro + haut-parleur • Ports GPIO (x20) + bluetooth |
| Système d'exploitation | Zephyr OS |
| Langages | Bloc, Javascript, Python |
| Alimentation | <ul style="list-style-type: none"> • USB 5V • Piles (3V) |
| Accessoires | <ul style="list-style-type: none"> • Lumière : bande led / matrice led, feu tricolore • Moteur : servo-moteur, barrière • Feu tricolore • Capteurs : ultrasons... |







02 - Usages




Découvrir le développement

-  Le langage block pour **néophyte**
-  IDE très visuel
-  Accessible aux enfants
-  Documentation





Apprendre un langage informatique

-  Python
Javascript
Typescript
-  Strict minimum

Se perfectionner en algo

-  Environnement minimal
-  Faibles capacités (CPU + mémoire)
-  Entretien d'embauche

Expérimenter ses idées


-  Nombreux capteurs
-  Ports GPIO
-  Pas cher
-  Librairies disponibles



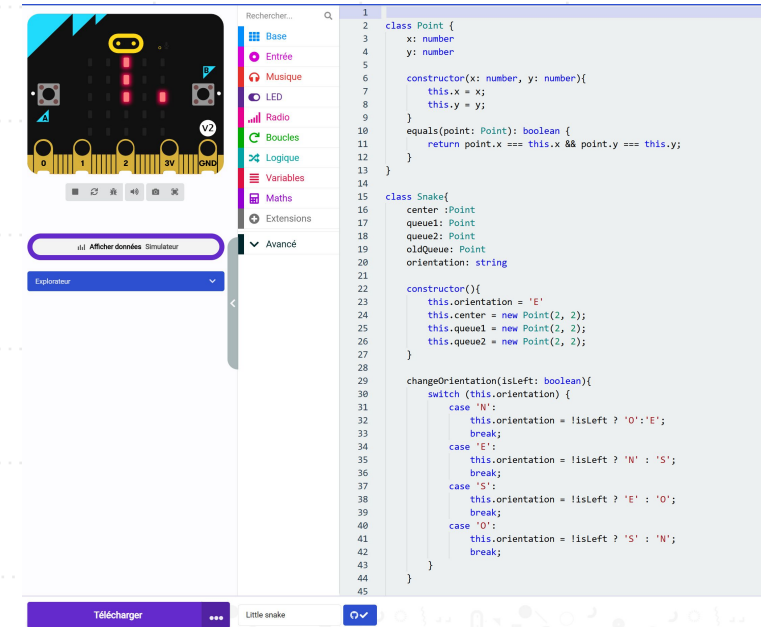
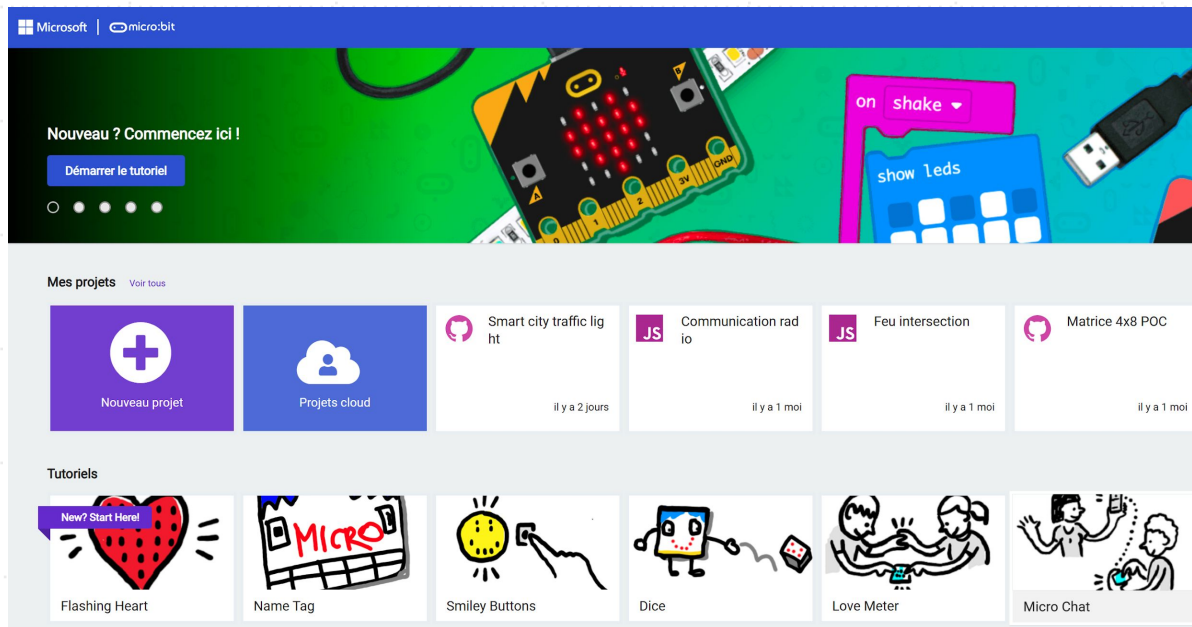
03 - Exercices

A vous de jouer...

La plateforme de développement

- <https://makecode.microbit.org>
- Compte  pour sauvegarder son code
- Simulateur de carte affiché pour tester le code
- Envoi du code sur la carte en USB

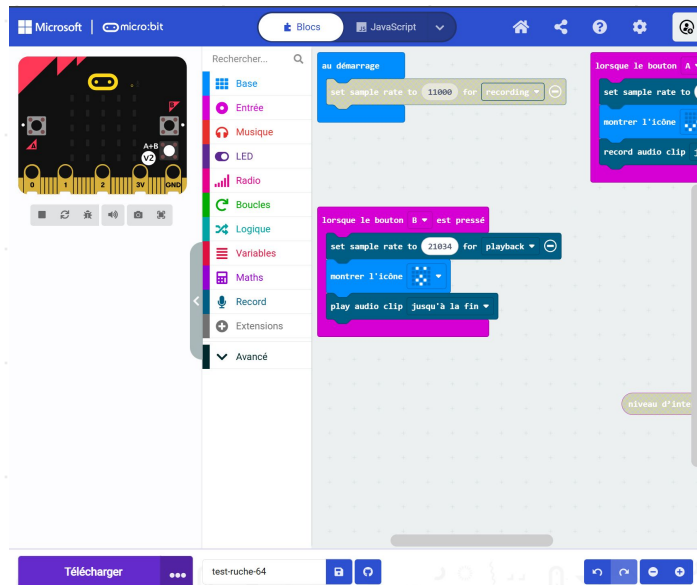
- Passage du bloc au javascript ou python
- Librairie de fonctions et auto-complétion
- Intégration de bibliothèques externes
- Création d'une librairie facilitée



Déployer sur la carte

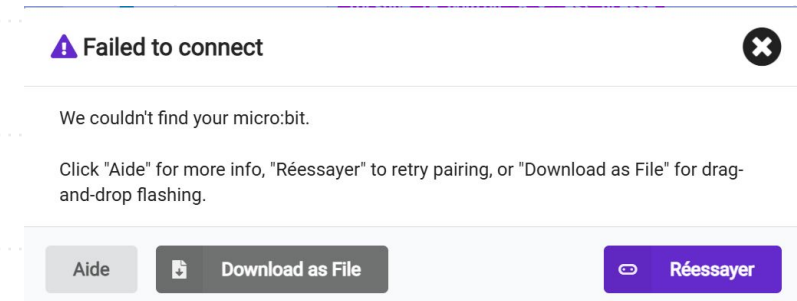
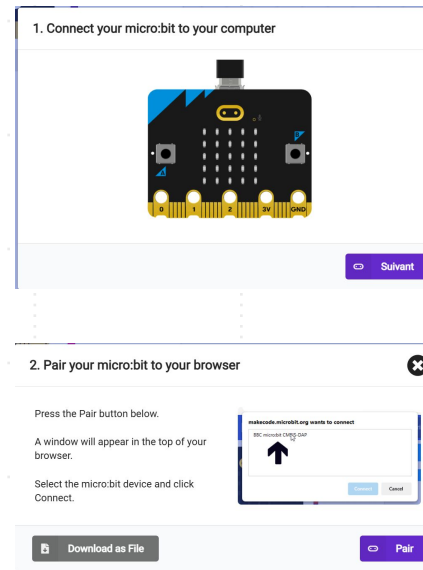
Windows

- Cliquer sur Télécharger
- Suivre les étapes
- Carte reconnue lors de l'appairage
- La carte est à jour



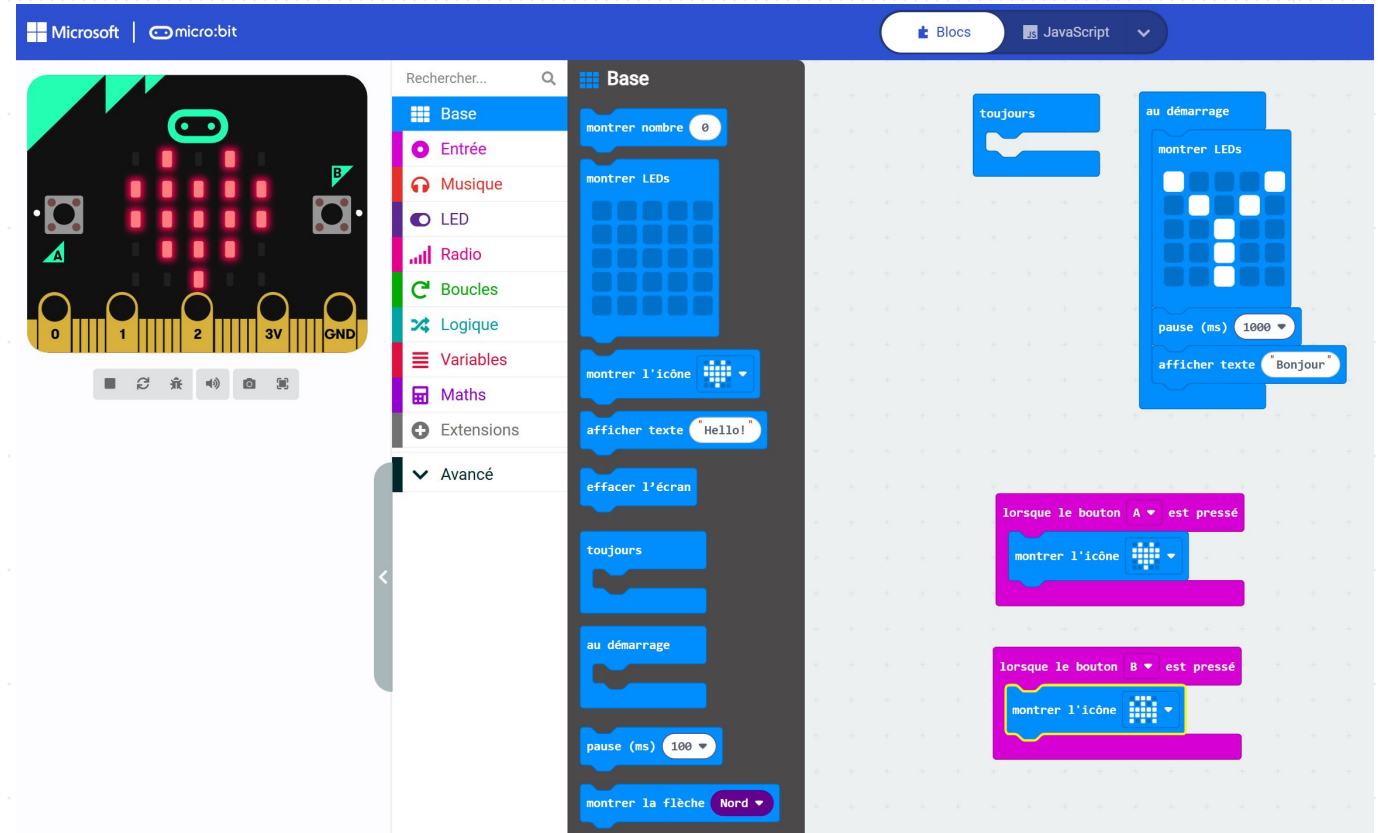
Linux

- Carte non reconnue par la plateforme
- Télécharger le binaire
- Le déplacer directement sur la carte (UMS)
- La carte redémarre et exécute le nouveau code



Découvrir la programmation

- **Langage** : bloc
- **Objectif** : manipuler du bloc comme un enfant
- **Exercice** :
 - Au démarrage, afficher une icône de cœur
 - Quand on appuie sur le bouton A, incrémenter un compteur et l'afficher
 - Quand on appuie sur le bouton B, réduire ce compteur et l'afficher
 - Quand on appuie sur les deux boutons, remettre le compteur à 0 et l'afficher



Programme

Découvrir la programmation 

Démo : faire des katas 

La bille qui roule 

Icônes par radio 

Tu chauffes tu brules  

Dépôt Github



<https://github.com/jotitan/microbit-volcamp-2025>

Debugger sur micro:bit

- Utilisation de `console.log(message)` pour tracer les messages :

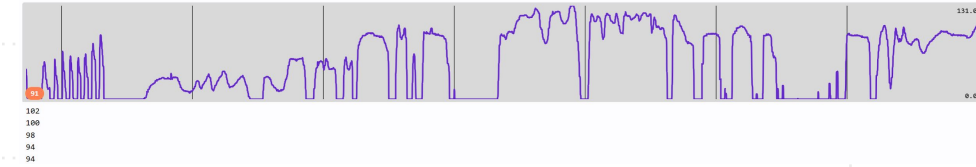
- Les afficher dans la console de déboguer classique (F12)

- Les afficher dans l'interface :

- soit les données du simulateur
- soit les données de la carte connectée

Afficher données Simulateur

Afficher données Appareil



- Mode debug :

- Quand ça plante, l'option « déboguer ce projet apparaît »
- Mode debug normal : points d'arrêt, pile d'exécution et variables

Problèmes 1


Cannot read properties of undefined (reading 'length')

à <main> (ligne 5)

Déboguer ce projet

```
Debug Mode [Quitter le mode Debug]
Variables
pressed: false
Pile d'exécution
→ <main> main.ts:0
1 let pressed = false
2 let counter = -1
3 //const rawSong = "C F F F G F F G A A B A G F F F F E D C C C F F G G F"
4 let rawSong:string;
5 const song = rawSong.split(" ")
6 basic.forever(function () {
7   if (!(pressed) && input.buttonIsPressed(Button.A)) {
8     pressed = true
9     counter = (counter + 1) % song.length
10    control.inBackground(function () {
11      const m = music.stringPlayable(song[counter], 200)
```

IOT : La bille qui roule

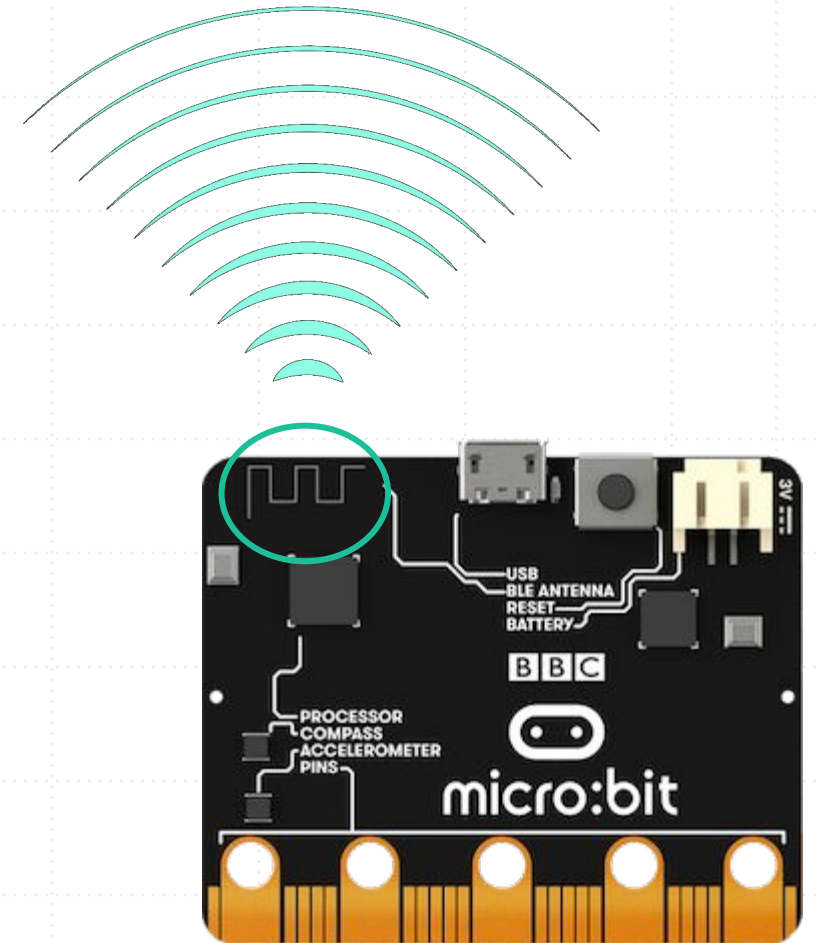
- **Langage** : Javascript
- **Objectif** : Simuler une bille sur un plateau qui bouge
- **Lien** :  <https://github.com/jotitan/microbit-rolling-ball>
- **Exercice** :
 - Utiliser la détection de l'orientation de la carte
 - Afficher sur la matrice de Led 5x5 la position de la bille
 - Bonus : modifier le pas




Radio sur micro:bit

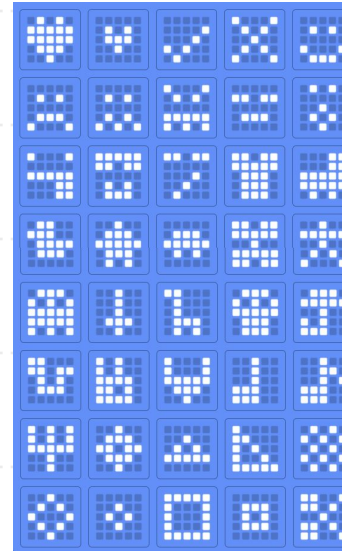
- Fonctionne en radio et bluetooth
- ⚠ Attention aux interférences
- Fréquence
 - Bande de fréquence : entre 2.4Ghz et 2.5Ghz
 - 84 pas de fréquences
 - `radio.setFrequency(frq) // 0-83`
- Group
 - Permet le filtrage des messages
 - 256 valeurs
 - `radio.setGroup(group); // 0-255`
- Fonctions de communication
 - Chaîne de caractères : 19 caractères max
 - Nombre : nombre décimal ou entier
 - Couple chaîne de caractères (8 caractères max) et un nombre
 - Buffer (19 octets max)
 - Emission : `sendXXXXXX` où xxxx est le type
 - Réception : `onReceivedXXXXXX` où xxxx est le type
 - Détail du paquet reçu : `receivedPacket`

Documentation : <https://makecode.microbit.org/reference/radio>



IOT : Communication radio

- **Langage** : Javascript
- **Objectif** : Faire communiquer les micro:bit par radio.
Chacun peut choisir son canal, son icône, et l'envoyer aux autres.
- **Lien** :  <https://github.com/jotitan/microbit-radio-icons>
- **Etapes** :
 - Sélection d'un canal de communication
 - Sélection d'une icône
 - Envoi de l'icône
 - Affichage d'une icône reçue sur le canal



IOT : Jeu du « Tu chauffes tu brules »

- **Langage** : Javascript
- **Objectif** : Trouver les balises cachées dans la salle
- **Lien** :  <https://github.com/jotitan/microbit-burn-cold-game>
- **Exercice** :
 - Plusieurs émetteurs sont présents dans la salle et envoient des messages sur les canaux 4 à 9 toutes les 500 ms
 - Ecrire un récepteur qui utilise la puissance de réception du message pour estimer relativement la distance
 - Afficher sur l'écran des indices pour évaluer la distance de la balise
 - Utiliser les boutons pour changer de groupe radio pour trouver les autres balises



Merci

