Né d'hier

Apprentissages bio-inspirés et robotique

Matière principale : Sciences et technologie, Mathématiques, Sciences humaines

Thématiques : Apprentissage automatique, algorithmes bio-inspirés, apprentissage par renforcement, adaptabilité, cognition humaine

Pratiques informatiques : Programmation par blocs, programmation tangible, découverte de l'IA

Activité débranchée rebranchée

Âge:
8-12 ans

Durée:
3h

Logistique : en classe, en groupe

Difficulté

Points du programme abordés

« Découvrir le monde du numérique » – Sciences et technologie, Cycle 3

Cette séquence permet aux élèves de découvrir les principes fondamentaux de l'apprentissage automatique en vivant eux-mêmes l'expérience d'un algorithme qui apprend par essai-erreur. En combinant une phase débranchée immersive, l'observation d'une IA qui apprend, et une phase de programmation robotique, les élèves comprennent comment les machines peuvent « apprendre » à résoudre des problèmes sans être explicitement programmées pour chaque situation. Les compétences mobilisées sont les suivantes :

- « Pratiquer des démarches scientifiques et technologiques » – Les élèves formulent des hypothèses sur les « bonnes routes », les vérifient par l'expérimentation, et découvrent le principe d'apprentissage par renforcement
- « Concevoir et produire un objet technique » – Programmation de robots pour réaliser un parcours en utilisant les connaissances acquises lors des phases précédentes
- « S'approprier des outils et des méthodes » – Création de « cartes mentales » du parcours, observation d'outils de simulation IA, travail collaboratif
- « Pratiquer des langages » – Expression des stratégies d'exploration, verbalisation des algorithmes découverts, comparaison humain/machine

Liens avec d'autres domaines du programme

- **Mathématiques** : Travail sur le repérage dans un quadrillage, les coordonnées, la logique conditionnelle (si... alors...). Notion intuitive de probabilité lors des choix d'exploration.
- **Géographie** : Le contexte de navigation permet d'aborder les thèmes de repérage spatial et de cartographie.
- **EMC (Enseignement Moral et Civique)** : Discussion sur l'éthique de l'IA : comment les voitures autonomes apprennent-elles ? Peut-on faire confiance à une machine qui apprend ?
- **Français** : Expression orale lors des phases de réflexion, argumentation pour justifier les choix stratégiques.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Matériel nécessaire

- **Grille 6×6** : Plusieurs options selon vos moyens et votre espace :
  - Version géante au sol : Tracez une grille avec du scotch de couleur (cases de ~30-40 cm). Vous pouvez même utiliser un des robots comme « pion » que les élèves déplacent !
  - Version papier A3/A4 : Imprimez les grilles fournies en annexe ou créez les votres. Utilisez des pions détournés de jeux de société (échecs, dames, Monopoly…) pour simuler les déplacements de l'agent autonome.
- **Fiches de mémorisation** : Une par équipe pour noter les découvertes (cases sûres, obstacles) – modèle fourni en annexe
- **Récompenses** : Bonbons, pompos, jetons ou points … pour simuler le renforcement positif et négatif
- **Ordinateur/tablette** : Accès à l'outil BioLearningGame développé dans le cadre du projet Erasmus + SteamCity : https://steamcity.github.io/BioLearningGame/ - Onglet “AI)
- **Vidéoprojecteur** : Pour démonstration collective de l'apprentissage de l'IA
- **Robots pour la partie programmation** : Si possible, les élèves tournent sur 3 ateliers robotiques, chacun proposant une logique de programmation différente avec ses défis spécifiques. Nous vous conseillons les outils suivants pour avoir une vraie expérience diversifiée de programmation :

Atelier	Robot	Logique de programmation	Défi spécifique
Atelier A	Cubetto	Programmation tangible : blocs physiques à placer sur un panneau	Optimiser avec un nombre limité de blocs (12 max)
Atelier B	Sphero Indi	Programmation visuelle par couleurs : tuiles colorées que le robot lit	Comprendre les comportements conditionnels (SI couleur ALORS action)
Atelier C	Robot micro:bit (Maqueen, Move Mini…) -	Programmation par blocs sur écran (MakeCode) - Exemple de programme en annexe	Calibrer précisément les mouvements, utiliser les capteurs

Adaptations aux contraintes matérielles

- *Avec 1 seul robot : L'activité fonctionne parfaitement avec un seul type de robot. Choisissez celui qui correspond à votre budget et vos objectifs. Sphero Indi est très accessibles pour débuter ; le robot micro:bit offre plus de possibilités mais demande un peu plus de prise en main. Cubetto est intéressant car il propose un défi proche du casse-tête pour les élèves, mais il est plus honéreux, et peu réutilisable car conçu pour les enfants de très jeune âge.*
- *Avec 2 robots : Vous pouvez comparer deux logiques de programmation différentes, ce qui est déjà très riche pédagogiquement.*
- *Pas de robot ? Rapprochez-vous des associations de médiation scientifique de votre territoire (Petits Débrouillards, fablabs, maisons pour la science, médiathèques, CANOPÉ…). Beaucoup proposent du prêt de matériel robotique ou peuvent intervenir dans votre classe avec leur équipement.*

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Introduction

Cette séquence constitue une activité introductive idéale à l'univers « Robots Meet Arts » (Rencontre Robots et Humanités). Elle permet de démystifier la programmation robotique de manière ludique, en montrant que programmer un robot n'est pas réservé aux experts : c'est accessible, créatif, et profondément connecté à ce que nous sommes en tant qu'êtres humains.

Le point de départ : Imaginez un robot qui doit apprendre à naviguer dans un environnement inconnu pour atteindre un objectif. Personne ne lui a donné de carte : il doit découvrir par lui-même quels chemins sont sûrs et lesquels sont bloqués par des obstacles. Comment peut-il apprendre sans se perdre définitivement ?

Dans cette séquence, les élèves vont d'abord jouer le rôle de ce robot explorateur, puis observer comment une véritable IA résout le même problème, et enfin programmer leurs propres robots pour réaliser un parcours sans faute. Ils découvriront qu'en explorant méthodiquement, en mémorisant ce qui fonctionne et ce qui ne fonctionne pas, ils construisent une « carte mentale » efficace du territoire – exactement comme le font les algorithmes d'apprentissage automatique quand ils construisent leur modèle.

L'originalité de cette activité réside dans son approche transdisciplinaire :

Discipline	Sciences du numérique	Sciences du vivant	Sciences humaines
Concepts clés	Programmation robotique, algorithmes, IA	Modèles bio-inspirés, modèle d'apprentissage des êtres biologiques	Psychologie de l'apprentissage, cognition humaine
Questions explorées	Comment programme-t-on un robot ?	Comment les êtres vivants apprennent-ils ?	Comment apprenons-nous en tant qu'être humain ?

L'apprentissage bio-inspiré - Quand les machines imitent le vivant

L'apprentissage est l'un des processus fondamentaux que partagent les humains et les machines, même si les mécanismes peuvent être très différents. Quand un enfant touche une plaque chaude et apprend à l'éviter, quand il découvre par essais successifs le meilleur chemin pour rentrer de l'école – il pratique l'apprentissage par renforcement. C'est exactement ce que font les algorithmes d'IA les plus puissants d'aujourd'hui.
L'objectif de cette activité est de fournir aux élèves une compréhension intuitive des modèles d'apprentissage bio-inspirés et de la façon dont les machines, à l'instar des organismes vivants, utilisent la méthode des essais et des erreurs pour s'adapter et trouver des solutions.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Introduction

Une entrée ludique dans la robotique

La robotique n'est pas qu'une affaire de code et de circuits. C'est aussi une invitation à questionner notre humanité : qu'est-ce qu'apprendre ? Qu'est-ce qui différencie l'intelligence humaine de l'intelligence artificielle ? Les robots peuvent-ils vraiment « comprendre » ou ne font-ils que calculer ? Ces questions philosophiques émergent naturellement de l'activité.

Cette activité a été conçue comme porte d'entrée accessible vers la programmation robotique. En commençant par une phase « débranchée » où les élèves sont eux-mêmes le robot, on lève les appréhensions : programmer, c'est simplement donner des instructions claires, comme celles qu'ils ont suivies pendant le jeu. Le passage aux vrais robots devient alors naturel et non intimidant.

Trois robots sont proposés (Cubetto, Sphero Indi, micro:bit), chacun avec une logique de programmation différente. L'idéal est de faire découvrir les trois en ateliers tournants, mais l'activité fonctionne très bien avec un seul robot selon vos moyens. L'essentiel est de vivre l'expérience de donner des instructions à une machine et de voir sa création prendre vie.

Liens utiles

- **BioLearningGame – Outil de simulation IA** : https://steamcity.github.io/BioLearningGame/
- **Machine Learning for Kids** : https://machinelearningforkids.co.uk
- **Cubetto – Documentation** : https://www.primotoys.com
- **Sphero Indi – Guide pédagogique** : https://sphero.com/pages/indi
- **MakeCode micro:bit** : https://makecode.microbit.org
- **KittenBot TabbyBot Programming Robotics Kit For micro:bit** – https://www.kittenbot.cc/products/kittenbot-tabbybot-programming-robotics-kit?_pos=4&_sid=ebad00185&_ss=r
- **Robot Maqueen pour Micro:bit** – https://www.kubii.com/fr/robots-extensions/2675-micro-bit-robot-maqueen-kubii-6959420915002.html?srsltid=AfmBOoqwqBgJywEXgdE4OmdAZZ5pqoZnWwtsurv0MqxIKJgCrqRBB1XB
- **CS Unplugged – Activités débranchées** : https://csunplugged.org

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Séquence 1 - Mise en route (Warm-up)

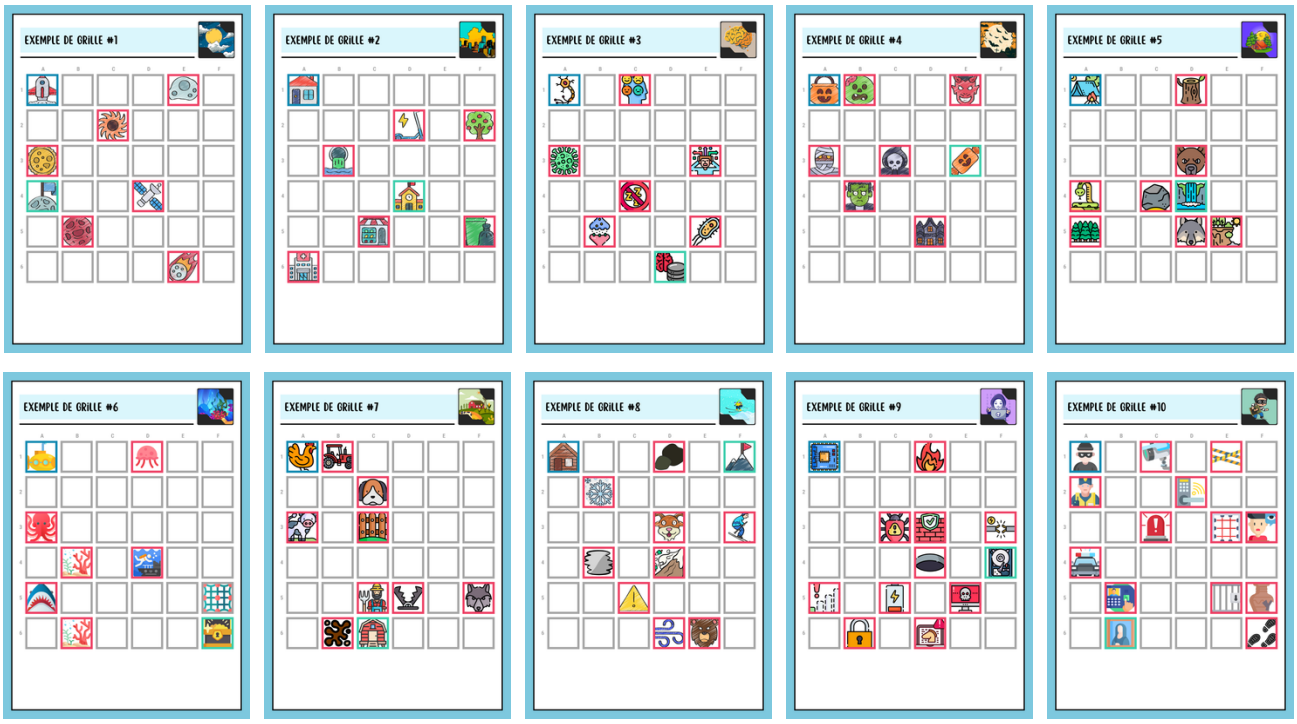
Découverte et échauffement

- Installer les grilles (2-3 selon effectif) – au sol ou sur tables
- Préparer les récompenses pour le renforcement (bonbons, jetons, pompons … selon votre approche)
- Imprimer et plastifier les fiches de mémorisation - les plastifier permet de les utiliser avec des feutres effaçables afin de les réutiliser ou de corriger les informations notées
- Avoir sous les yeux le plan de chaque grille (pour donner les retours)
- Prévoir un pion par groupe (figurine, robot éteint, ou élève-pion)

Notes pour l'enseignant·e

Objectif : Comprendre l'apprentissage par essais et erreurs, et construire son propre modèle d'apprentissage par renforcement.

Formez 2 ou 3 groupes selon l'effectif (idéalement 10-12 élèves par grille). Chaque groupe travaillera autour de sa propre grille. Si possible, créez les grilles au sol, car elles seront utiles lors des phases suivantes. Créez ou sélectionnez une grille qui correspond à vos objectifs : plus ou moins d'obstacles, zones inaccessibles, temps disponible … Voici quelques exemples disponibles pour impression en annexe :



Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Règles du jeu

Chaque groupe dispose d'une grille 6×6 et d'un pion (figurine, robot éteint, ou un élève qui se déplace). Le groupe décide collectivement des mouvements (haut, bas, gauche, droite - interdiction de se déplacer en diagonale), qui sont obligatoirement limités à une case par tour. Pour matérialiser l'apprentissage par renforcement, utilisez des jetons (ou bonbons, pions de couleur, haricots...). L'enseignant (ou un élève « maître du jeu ») donne un retour après chaque mouvement : soit un déplacement valide et le gain d'une récompense dans la cagnotte collective = renforcement positif, soit un déplacement interdit - l'enseignant retire alors la totalité de la cagnotte gagnée jusqu'alors et les élèves repartent du point de départ = renforcement négatif.

Organisation du jeu : Rassemblez chaque groupe autour de sa grille. Annoncez le contexte sans révéler l'objectif : « Vous êtes un robot qui doit naviguer dans cette grille. Votre mission : trouver un point précis. Le problème ? Vous ne savez pas où il est ! Vous devrez apprendre par vous-mêmes en explorant. ». Ne donnez aucune information sur l'objectif ou la disposition des obstacles. Les élèves doivent découvrir par eux-mêmes, exactement comme une machine qui apprend sans connaissances préalables.

Chaque élève reçoit une **fiche de mémorisation vierge** (grille vide ou feuille blanche). L'enseignant peut expliquer : « Vous pouvez utiliser cette fiche pour noter ce que vous découvrez. C'est votre fiche personnelle, vous pouvez la remplir comme vous le voulez. Attention : cette fiche sera votre seule aide pour programmer votre robot dans la suite de l'activité ! » L'exploration est collective (le groupe décide ensemble des mouvements), mais chaque élève remplit sa propre fiche de mémorisation individuelle. Cela permet d'observer la diversité des représentations mentales pour une même expérience vécue.

Point pédagogique – NE PAS GUIDER !

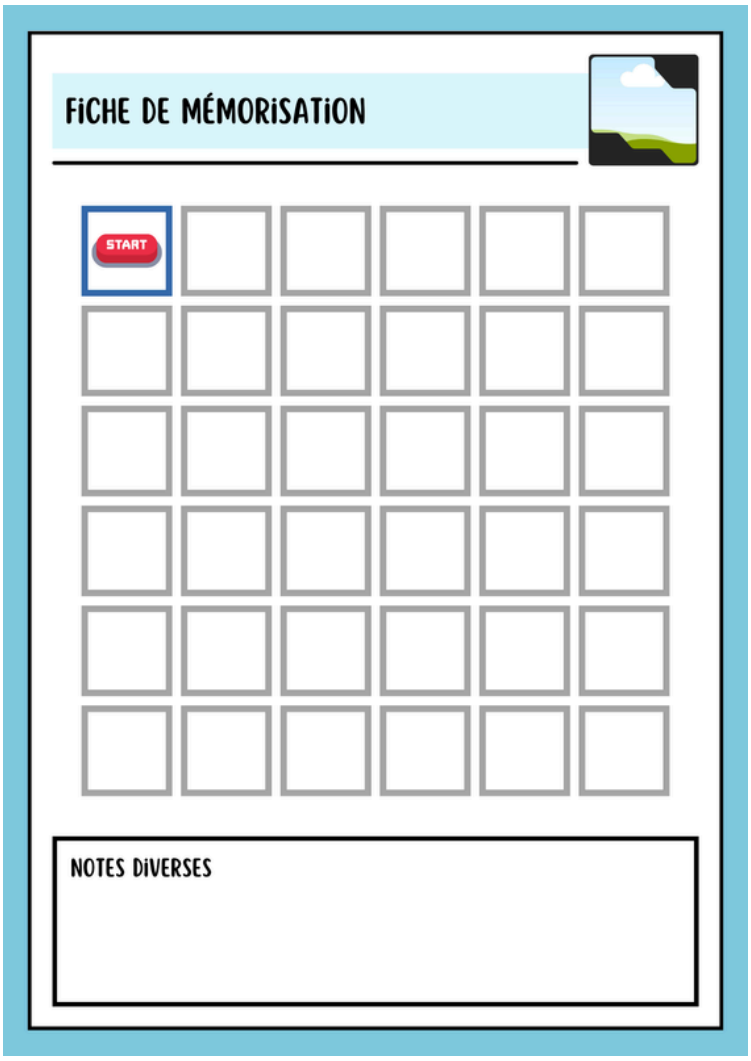
Ne donnez AUCUNE consigne sur comment remplir la fiche. Pas de modèle, pas d'exemple, pas de suggestion. L'objectif est d'observer comment chaque élève construit spontanément son propre système de représentation : certains dessineront des flèches pour indiquer le chemin, d'autres mettront des croix ou des X sur les obstacles, d'autres utiliseront des couleurs (vert = sûr, rouge = danger) … Cette diversité EST l'apprentissage. Même si tous les élèves vivent la même exploration collective, chacun construit son propre « modèle » avec sa propre logique. Cela est également valide pour les déplacements sur la grille. Ne donnez aucune indication du nombre d'obstacle ou d'indice sur le positionnement de l'objectif. Par exemple, certains groupes d'élèves présupposent que l'objectif est dans un coin de la carte, sur une case spécifique … il est intéressant d'observer ces biais et d'en discuter une fois l'activité terminée.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Une fois que les groupes ont trouvé le chemin, comptez les jetons ! Introduisez le vocabulaire scientifique. Le jeton gagné correspond au renforcement positif. Le jeton perdu correspond aux points perdus : c'est le renforcement négatif. Leur fiche de mémorisation correspond à un modèle mental. Ce type d'apprentissage s'appelle l'apprentissage par renforcement – c'est comme ça qu'on entraîne les voitures autonomes, les robots, et même les IA qui jouent aux jeux vidéo.

Chaque élève garde sa fiche de mémorisation.

Chaque élève garde sa fiche de mémorisation. L'enseignant peut annoncer la transition vers la phase suivante en expliquant que les élèves vont maintenant apprendre à programmer un robot, et que plus tard, ils devront le programmer pour qu'il suive le chemin découvert, en s'appuyant uniquement sur leur fiche de mémorisation.



Fin de la séquence de mise en route

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Séquence 2 - Approfondissement (Build-up)

Acquisition et structuration des savoirs

- Vérifier que les robots sont chargés et fonctionnels
- Préparer l'espace de travail (zone dégagée au sol)
- Imprimer les fiches missions correspondant au(x) robot(s) utilisé(s)
- Avoir lu le tutoriel enseignant du/des robot(s)
- Si la phase 1 de warm-up a été réalisée sur des grilles papier, créer une version plus grande de la grille au sol (scotch, 30-40 cm/case)

Notes pour l'enseignant·e

Objectif : Maîtriser les bases de la programmation robotique à travers des missions progressives.

Cett étape est une phase de découverte technique volontairement déconnectée du défi principal. Les élèves apprennent à manipuler le robot sans pression, à travers des missions progressives qui n'ont rien à voir avec la grille du Warm-up. **Cette séparation est intentionnelle :**

- Éviter la surcharge cognitive : apprendre un nouvel outil ET résoudre un problème complexe en même temps serait trop difficile
- Construire la confiance : les élèves arrivent à la phase finale en maîtrisant déjà leur robot
- Permettre l'erreur : pendant cette étape, se tromper n'a pas de conséquence sur le défi final

Quelle que soit le robot utilisé, les missions progressives permettent de découvrir :

1. **Le principe de base** : comment donner une instruction au robot
2. **Les déplacements** : avancer, tourner
3. **L'enchaînement** : combiner plusieurs instructions en séquence
4. **Les contraintes** : chaque robot a ses limites et sa logique propre

Les élèves découvrent également qu'un robot n'interprète pas les intentions – il exécute exactement ce qu'on lui dit. Si le robot fait quelque chose d'inattendu, c'est que l'instruction n'était pas celle qu'on pensait donner. Cette prise de conscience est fondamentale pour la suite.

Si vous pouvez avoir accès aux trois robots proposés, organisez des ateliers tournants. Chaque groupe découvre successivement différentes façons de programmer. Si vous n'avez accès qu'à un seul robot, vous pouvez organiser d'autres ateliers en parallèle, sur un temps dédié au numérique ou à d'autres activités créatives, afin d'éviter la surcharge d'élèves autour d'un seul outil technique.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

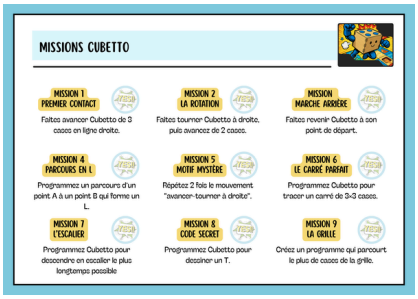
L'ordre des ateliers entre les différents robots n'a pas d'importance, car ce qui compte, c'est de vivre l'expérience de différentes logiques de programmation avec plusieurs objectifs pédagogiques :

- Déconstruire l'idée qu'il n'y a qu'une seule façon de programmer
- Comprendre que "programmer" = donner des instructions, quelle que soit la forme (blocs, tuiles, code)
- Enrichir la discussion : « Quel robot avez-vous préféré ? Pourquoi ? » révèle des profils d'apprentissage différents

Il n'y a pas de hiérarchie ! Cubetto n'est pas « plus facile » que micro:bit, ils sont différents. Chaque robot a sa logique propre, et maîtriser l'un n'aide pas forcément pour l'autre. C'est cette diversité qui fait la richesse de l'activité.

Robot	Programmation	Contraintes
Cubetto	Blocs physiques en bois à placer sur un panneau – programmation tangible	Nombre de blocs limité (12 max) → oblige à planifier et optimiser
Sphero Indi	Tuiles colorées posées au sol – programmation visuelle par couleurs	Dépend des tuiles disponibles, comportements prédéfinis par couleur
Robot micro:bit	Blocs sur écran (MakeCode) puis téléchargement – programmation par bloc	Calibrage des durées et distances

Laisser explorer les élèves grâces aux fiches missions. Certaines missions nécessitent l'utilisation de la grille 6x6 découverte lors du warm-up afin de préparer la mission finale i.e. créer un programme qui permet au robot autonomement de réaliser le parcours découvert en phase 1. Si vous avez utilisé des grilles papier pendant le Warm-up, vous devrez créer une grille au sol de 6x6 graĉe à du scotch de couleur, des cases de 30-40 cm adaptées à la taille du robot. Si vous aviez déjà utilisé une grille au sol pour la phase 1, ne la retirez pas avant la fin complète de l'activité. Dans les robots proposés, seul Cubetto est livré avec son propre tapis 6×6 parfaitement calibré pour ses déplacements. Utilisez directement ce tapis car le Cubetto ne peut pas être calibré en terme de longueur de déplacement !



Pendant que les élèves réalisent les missions, tournez entre les groupes et valorisez les erreurs : « Ton robot a tourné dans le mauvais sens ? Super, tu viens d'apprendre quelque chose ! ». Observez les stratégies : qui planifie ? qui teste directement ? qui collabore ?

Fin de la séquence d'approfondissement

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Séquence 3 - Mise en pratique (Rehearsal)

Réinvestissement et application des connaissances

Nous vous conseillons de réaliser cette phase en parallèle de la phase 2. Le mieux est de proposer cette étape comme mission finale dans la continuité des fiches missions de l'étape précédente. Ainsi, chaque groupe suit le cycle suivant :

- Atelier Robot A : Missions → Défi Rehearsal → Rotation
- Atelier Robot B : Missions → Défi Rehearsal → Rotation
- Atelier Robot C : Missions → Défi Rehearsal → Rotation

Notes pour l'enseignant·e

Objectif : Utiliser sa fiche de mémorisation pour programmer le robot à suivre le chemin découvert pendant le Warm-up.

Afin de conclure l'activité, les élèves doivent programmer leur robot pour qu'il suive le chemin qu'ils ont découvert pendant le Warm-up. La règle d'or : **Ils ont UNIQUEMENT leur fiche de mémorisation**. Ils ne peuvent pas voir la grille complète avec ses obstacles avant d'avoir programmé ! Les élèves étudient leur fiche de mémorisation et planifient leur programme :

- Quel chemin avaient-ils trouvé ?
- Quelles instructions donner au robot ?
- Dans quel ordre ?

C'est ici que la qualité de la fiche de mémorisation se révèle ! Certains élèves auront noté clairement le chemin, d'autres devront « décoder » leurs propres notes.
Les élèves programment leur robot en se basant uniquement sur leur fiche. Ils n'auront pas le droit de toucher à leur robot une fois leur programme lancé, il faut donc qu'il parcourt la grille autonomement.

Robot	Comment programmer le chemin
Cubetto	Placer les blocs dans l'ordre du chemin mémorisé (sur le tapis 6×6 fourni)
Sphero Indi	Disposer les tuiles pour créer le parcours mémorisé
micro:bit	Coder la séquence de mouvements dans MakeCode

Pour valider leur programme, le groupe pose son robot sur le départ de la grille, lance le programme et la classe observe si leur agent suit le bon chemin et attent l'objectif. Si le robot échoue, laissez le groupe corriger son programme et tester à nouveau.
En programmant le même chemin avec différents robots, les élèves réalisent que la même logique (leur fiche) s'exprime différemment selon l'outil et certains robots sont plus adaptés à certains parcours. Leur fiche de mémorisation doit être suffisamment claire pour être "traduite" dans n'importe quel langage

Fin de la séquence de mise en pratique

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Réflexion autour de la séquence

Conclure et en tirer des apprentissages

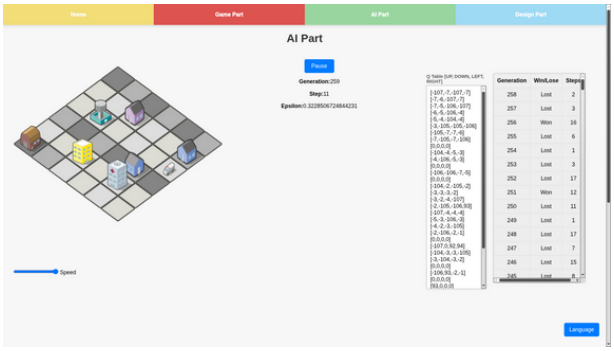
C'est le moment de partager et comparer les différents modèles créés pendant l'activité et de comprendre nos modes d'apprentissage en les comparant à ceux d'une machine.

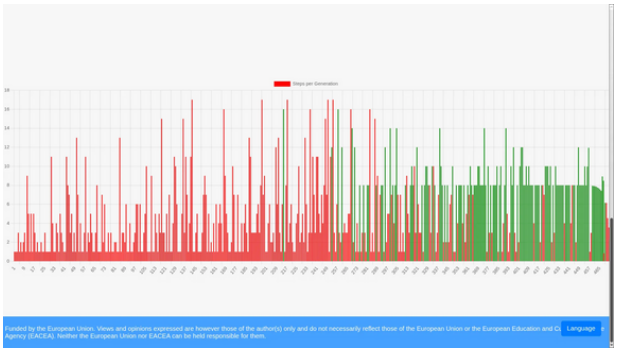
Affichez les fiches (ou faites circuler) et animez une discussion :

- « Quels systèmes de notation avez-vous inventés ? »
- « Lesquels ont bien fonctionné pour programmer le robot ? Pourquoi ? »
- « Y a-t-il eu des difficultés à relire vos propres notes ? »
- « Si vous deviez refaire l'activité, que changeriez-vous sur votre fiche ? »

Maintenant que les élèves ont vécu l'expérience d'apprendre un chemin et de le mémoriser, montrez-leur comment une intelligence artificielle résout exactement le même problème.

Projetez BioLearningGame (https://steamcity.github.io/BioLearningGame/), spécifiquement l'onglet appelé "AI Part". Les élèves observent qu'au début, l'IA se déplace au hasard, comme eux au début de la première phase. Elle fait beaucoup plus d'erreurs que les humains : elle sort de la grille, retourne en arrière, tourne en rond. Elle teste la grille comme un labyrinthe – pour elle, une case peut être un obstacle si on y entre par la gauche, mais pas si on y entre par le haut ! Petit à petit, elle évite les obstacles qu'elle a rencontrés, et finit par trouver le chemin optimal.





Les chiffres surprennent : l'IA peut mettre jusqu'à 700 essais pour trouver le chemin optimal, mais elle fait ces 700 essais en moins d'une minute. Les élèves, eux, ont trouvé en quelques essais... mais ça leur a pris beaucoup plus de temps. L'IA compense son ignorance de l'environnement par sa vitesse.

Observez aussi ce que l'IA cherche à optimiser : elle veut le meilleur score possible, c'est-à-dire atteindre l'objectif avec le minimum de pas. Elle ne cherche pas juste « un chemin qui marche », elle cherche « le meilleur chemin ». C'est pour ça qu'elle continue à explorer même après avoir trouvé une solution.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Réflexion autour de la séquence

Conclure et en tirer des apprentissages

Pendant l'observation, posez des questions : Comment l'IA se déplace-t-elle au début ? (Au hasard, elle sort même de la grille !) Pourquoi fait-elle autant d'erreurs ? (Elle n'a pas d'intuition, elle teste TOUT) Combien de tentatives lui faut-il ? (Des centaines !) Pourquoi continue-t-elle après avoir trouvé un chemin ? (Elle cherche le chemin optimal, le plus court)

Animez ensuite une discussion comparative. Les élèves ont trouvé le chemin en quelques essais, l'IA en 700. Les élèves ont mis plusieurs minutes, l'IA moins d'une minute. Les élèves réfléchissent avant d'agir, l'IA essaie tout sans réfléchir. Les élèves restent dans la grille (logique !), l'IA sort de la grille et teste tout. Les élèves utilisent leur intuition, l'IA n'a que des calculs. Les élèves cherchent « un chemin qui marche », l'IA cherche « le chemin optimal ». Les élèves peuvent expliquer leur stratégie, l'IA ne peut pas. Les élèves s'adaptent facilement à un nouveau problème, l'IA doit tout réapprendre depuis zéro.

Posez les questions de discussion : Qui a appris plus vite, vous ou l'IA ? (Ça dépend comment on compte ! Moins d'essais pour vous, moins de temps pour l'IA) Quels sont les avantages de l'IA ? (Ne se fatigue pas, très rapide, trouve la solution optimale, ne se décourage pas) Quels sont VOS avantages ? (Intuition, peu d'erreurs, capacité à expliquer, à s'adapter, à comprendre) Dans quels cas l'IA serait-elle meilleure ? Et vous ?

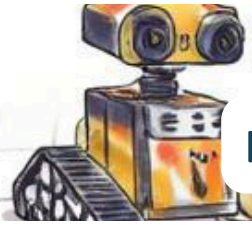
Concluez sur la richesse de ce qu'ils ont vécu : « Aujourd'hui, vous avez fait bien plus que programmer un robot. Vous avez découvert comment VOUS apprenez, vous avez inventé votre propre système de notation, et vous avez compris comment les machines s'inspirent de l'apprentissage du vivant. »

Ouvrez sur des questions plus larges : Où retrouve-t-on ce type d'apprentissage dans la vie quotidienne ? (GPS, jeux vidéo, robots aspirateurs...) Les robots peuvent-ils vraiment « comprendre » ou ne font-ils que calculer ? Pour la trace écrite, créez ensemble une affiche ou un document récapitulatif avec les différents systèmes de notation inventés par la classe, les mots-clés (apprentissage par renforcement, modèle mental, essai-erreur), un schéma comparatif humain/machine, et les réflexions des élèves.

Fin de la séquence "Né d'hier"

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Ressources imprimables



Fiche de mémorisation

START

Notes diverses

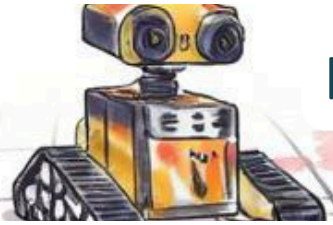
Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

Cofinancé par l'Union européenne

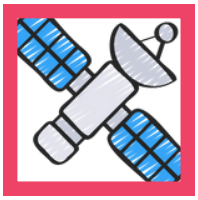
Consortium:

LAB · UCLL · twa. · STIMMULI for social change · UNIVERSITAT ROVIRA I VIRGILI · de ark basisschool

Ressources imprimables



Exemple de grille #1

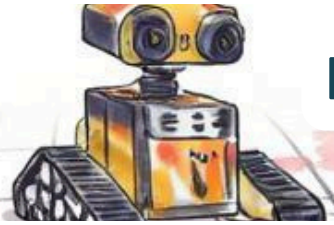
	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

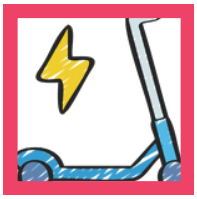
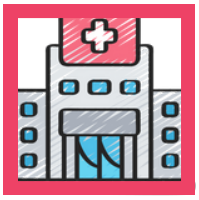
Cofinancé par l'Union européenne

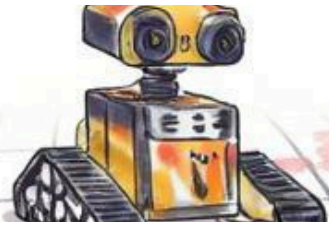
Consortium:

Ressources imprimables



Exemple de grille #2

	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

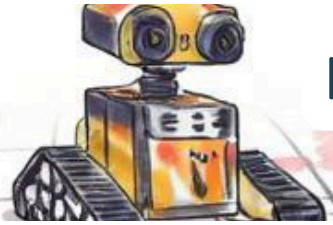


Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

Cofinancé par l'Union européenne

Consortium:

Ressources imprimables



Exemple de grille #3

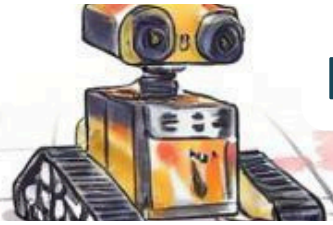
	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

Cofinancé par l'Union européenne

Consortium:

Ressources imprimables



Exemple de grille #4

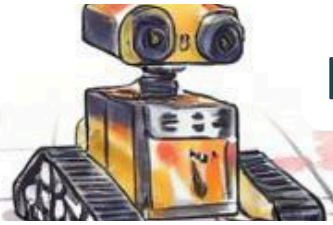
	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

Cofinancé par l'Union européenne

Consortium:

Ressources imprimables



Exemple de grille #5

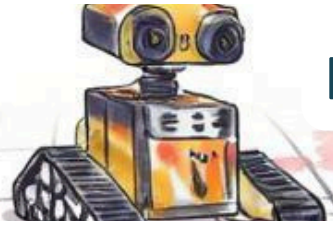
	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

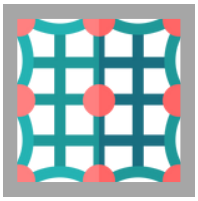
Cofinancé par l'Union européenne

Consortium:

Ressources imprimables



Exemple de grille #6

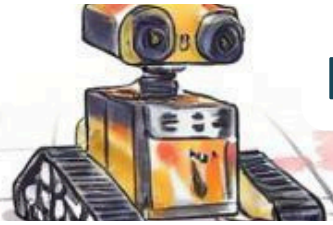
	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

Cofinancé par l'Union européenne

Consortium:

Ressources imprimables



Exemple de grille #7

	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

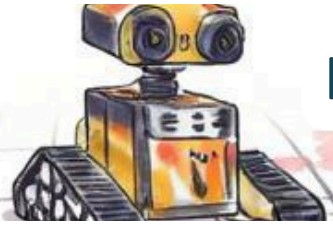
Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

Cofinancé par l'Union européenne

Consortium:

robots meet ARTS

Ressources imprimables



Exemple de grille #8

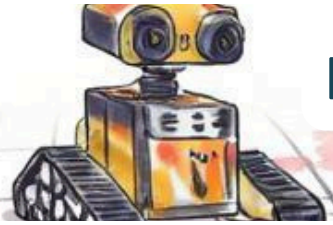
	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

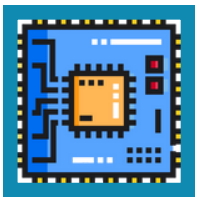
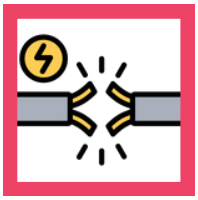
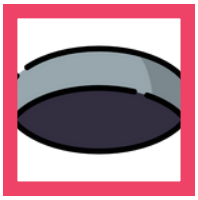
Cofinancé par l'Union européenne

Consortium:

Ressources imprimables



Exemple de grille #9

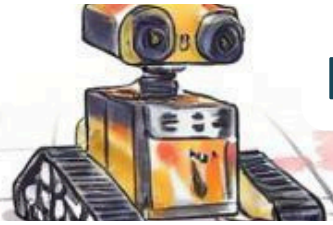
	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

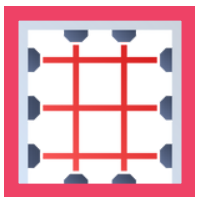
Cofinancé par l'Union européenne

Consortium:

Ressources imprimables



Exemple de grille #10

	A	B	C	D	E	F
1						
2						
3						
4						
5						
6						

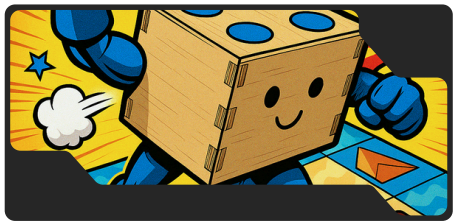
Numéro de projet : 2023-1-FR01-KA220-SCH-000151881

Cofinancé par l'Union européenne

Consortium:

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

MISSIONS CUBETTO



MISSION 1 PREMIER CONTACT

Faites avancer Cubetto de 3 cases en ligne droite.

MISSION 2 LA ROTATION

Faites tourner Cubetto à droite, puis avancez de 2 cases.

MISSION MARCHE ARRIÈRE

Faites revenir Cubetto à son point de départ.

MISSION 4 PARCOURS EN L

Programmez un parcours d'un point A à un point B qui forme un L.

MISSION 5 MOTIF MYSTÈRE

Répétez 2 fois le mouvement "avancer-tourner à droite".

MISSION 6 LE CARRÉ PARFAIT

Programmez Cubetto pour tracer un carré de 3×3 cases.

MISSION 7 L'ESCALIER

Programmez Cubetto pour descendre en escalier le plus longtemps possible

MISSION 8 CODE SECRET

Programmez Cubetto pour dessiner un T.

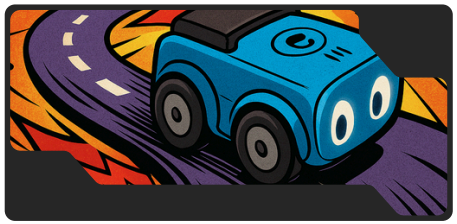
MISSION 9 LA GRILLE

Créez un programme qui parcourt le plus de cases de la grille.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Ressources imprimables

MISSIONS SPHERO INDI



MISSION 1 LIGNE DROITE

Faites avancer Cubetto de 3 cases en ligne droite.

MISSION 2 ALLER-RETOUR

Faites avancer Indi puis faites-le revenir à son point de départ.

MISSION VIRAGE À DROITE

Faites tourner Indi à 90° vers la droite puis avancez.

MISSION 4 STOP !

Faites avancer Indi puis arrêtez-le précisément.

MISSION 5 VITESSE VARIABLE

Créez un parcours avec changements de vitesse.

MISSION 6 SLALOM FLUIDE

Slalomez sur un parcours à 3 obstacles, dont le point de départ et d'arrivée se situe sur la même route.

MISSION 7 LE PARCOURS EN S

Créez un parcous en S sur une zone minimale d'un m^2.

MISSION 8 L'ÉTOILE

Sur grille : partez du centre et visiter les 4 bords de la grille.

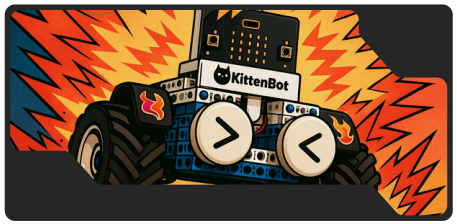
MISSION 9 LA DIAGONALE

Sur grille : allez du coin bas-gauche au coin haut-droit.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Ressources imprimables

MISSIONS MICRO:BIT



robots meet ARTS

MISSION 1 HELLO ROBOT

Faites avancer le robot sur une ligne droite d'au moins 2 mètres.

MISSION 2 ROBOT BALLET

Faites danser le robot sur place (de manière gracieuse).

MISSION CALIBRAGE

Trouvez comment faire avancer votre robot d'une case seulement sur la grille.

MISSION 4 LES VIRAGES

Faites un parcours sur la grille : avance, tourne à droite, avance, tourne à gauche.

MISSION 5 LE L SUR GRILLE

Sur la grille : 3 cases avant, droite, 2 cases avant.

MISSION 6 LE CARRÉ

Sur grille : réalisez un carré de 3×3 cases en utilisant une boucle.

MISSION 7 LES BOUCLES

Faites avancer le robot 6 fois avec seulement 2 lignes de code.

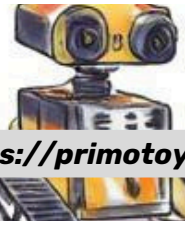
MISSION 8 LES FONCTIONS

Réalisez la mission précédente en utilisant les fonctions.

MISSION 9 L'AFFICHAGE

Ajoutez des messages grâce aux LED et des sons pendant les déplacements.

Tutoriel



Cubetto

Source : https://primotoys.com/cubetto-user-manual/

Introduction

Cubetto est une interface de programmation tangible conçue pour initier les enfants de 3 à 6 ans à la pensée computationnelle et à la logique de programmation. Il permet de programmer un petit robot en bois à l'aide de blocs colorés.
Les enfants manipulent la « file d'instructions », l'un des concepts fondamentaux de la programmation, sans avoir besoin de savoir lire ni d'utiliser d'écran.

Le kit Cubetto

Le robot : Cubetto est un robot cubique en bois avec un sourire gravé sur sa face avant. Il se déplace sur deux roues avec deux supports plastiques fixes (appelés « roulettes ») pour assurer sa stabilité. Caractéristiques principales :

- Avance de 15 cm exactement par instruction
- Effectue une rotation de 90° exactement lors des virages
- Joue une mélodie unique pour chaque type de commande
- Joue une mélodie plus longue lorsqu'une séquence est terminée

Le tableau d'interface : Le tableau d'interface est l'endroit où les enfants créent leurs programmes en insérant des blocs d'instructions dans des emplacements. On distingue :

- Séquence principale (emplacements 1 à 12) : Reliés par une ligne gravée, ces emplacements accueillent le programme principal
- Ligne de fonction (emplacements F1 à F4) : Dans une zone séparée, utilisée pour créer des sous-programmes réutilisables

Les blocs d'instructions : Quatre types de blocs en bois colorés, chacun représentant une commande spécifique :

Bloc	Couleur	Forme	Action
Avant	Vert	Flèche vers le haut	Cubetto avance de 15 cm
Gauche	Jaune	Flèche vers la gauche	Cubetto pivote de 90° dans le sens antihoraire
Droite	Rouge	Flèche vers la droite	Cubetto pivote de 90° dans le sens horaire
Fonction	Bleu	Ligne ondulée	Cubetto exécute la ligne de fonction

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Tous les blocs ont une base en forme de D qui s'insère dans les fentes du tableau, garantissant qu'ils ne peuvent être insérés que dans un seul sens.

Le tapis de jeu : Un tapis de jeu de 1 m × 1 m représentant « Le monde de Cubetto » – une grille de 6×6 cases de 15 cm × 15 cm. Chaque case correspond exactement à un mouvement avant de Cubetto.

Mise en route

Étape 1 : Insérer les piles

Cubetto et le tableau d'interface nécessitent chacun 3 piles AA.

Étape 2 : Mise sous tension

1. Trouver l'interrupteur coulissant sous chaque appareil
2. Le faire glisser sur la position ON
3. Cubetto : joue une mélodie d'accueil à la mise sous tension
4. Tableau d'interface : toutes les LED de retour clignotent trois fois à la mise sous tension

Étape 3 : Appairer les appareils

Cubetto et le tableau d'interface communiquent sans fil et doivent être appairés avant utilisation.

1. Allumer d'abord le tableau d'interface, puis Cubetto
2. Insérer n'importe quel bloc de direction (vert, jaune ou rouge – pas bleu) dans le premier emplacement
3. Appuyer sur le bouton d'action
4. Cubetto devrait exécuter l'instruction
5. Les appareils sont maintenant appairés !

Note : L'appairage s'effectue automatiquement en arrière-plan. La portée sans fil en intérieur est d'environ 10 mètres. Plusieurs kits Cubetto peuvent être utilisés à proximité sans interférence.

En cas d'échec de l'appairage : Éteindre les deux appareils, attendre quelques secondes, puis répéter la procédure.

Programmer Cubetto

1. Planifier l'itinéraire : Observer le tapis et décider où faire aller Cubetto
2. Choisir les blocs : Sélectionner les blocs d'instructions nécessaires
3. Construire la séquence : Insérer les blocs dans le tableau d'interface, en commençant par l'emplacement 1
4. Vérifier les LED : Chaque bloc correctement inséré allume sa LED de retour
5. Appuyer sur GO ! : Appuyer sur le bouton d'action pour envoyer le programme à Cubetto
6. Observer et apprendre : Cubetto exécute chaque instruction dans l'ordre

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Règles importantes

- Les séquences commencent à l'emplacement 1 et suivent la ligne gravée sur le tableau
- Pas de trous : Les blocs insérés après un emplacement vide ne seront pas reconnus
- Un programme à la fois : Modifier les blocs pendant l'exécution n'a aucun effet – attendre que Cubetto ait terminé
- Les LED de retour clignotent pour indiquer quelle instruction est en cours d'exécution
- Mélodies uniques : Cubetto joue un son différent pour chaque type de commande, aidant les enfants à suivre

Utiliser le bloc fonction

Le bloc fonction bleu est un outil puissant qui permet aux enfants de regrouper plusieurs instructions dans un seul bloc. Cela introduit le concept de fonctions (ou sous-programmes) – une idée fondamentale en programmation.

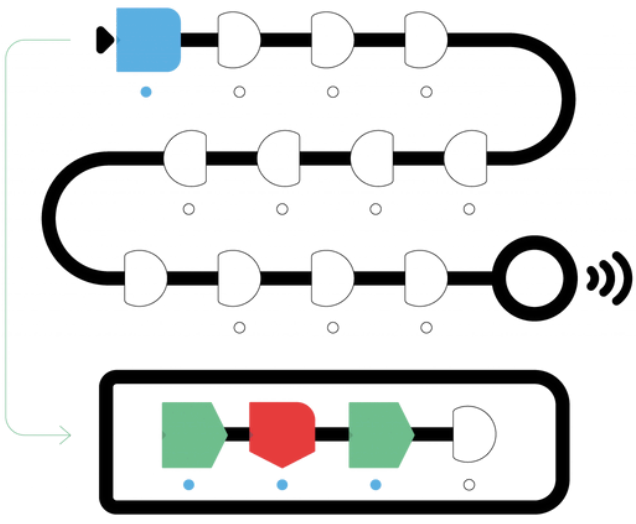
Comment fonctionnent les fonctions

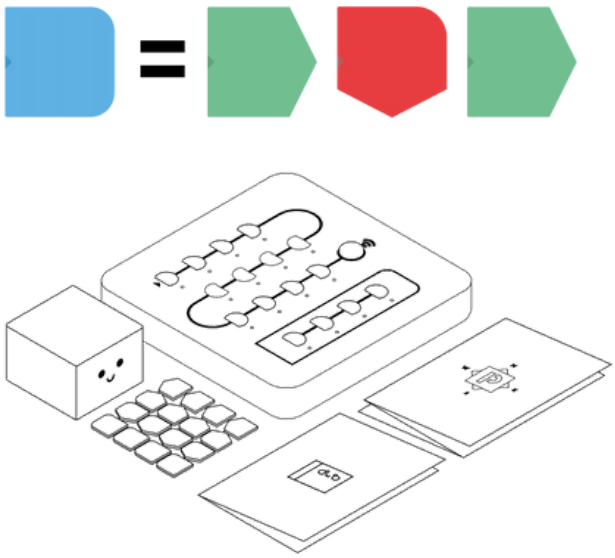
1. Créer une fonction : Insérer une séquence de blocs dans la ligne de fonction (les quatre emplacements en bas du tableau)
2. Appeler la fonction : Placer le bloc fonction bleu n'importe où dans la séquence principale
3. Exécution : Lorsque Cubetto rencontre le bloc bleu, il exécute toutes les instructions de la ligne de fonction, puis continue avec la séquence principale

Pourquoi les fonctions sont importantes

Les fonctions apprennent aux enfants à :

- Identifier des motifs : « Je fais la même chose encore et encore »
- Optimiser : « Comment puis-je faire cela avec moins de blocs ? »
- Penser de manière abstraite : « Ce bloc bleu signifie 'faire le virage' »
- Résoudre des problèmes complexes : Avec seulement 12 emplacements principaux, les fonctions permettent des programmes plus longs





Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Tutoriel

Sphero Indi

Source : https://cdn.shopify.com/s/files/1/0306/6419/6141/files/Indi-Booklet_150x150_v15_outlined.pdf

Introduction

Sphero Indi est un robot programmable conçu pour les jeunes apprenants (à partir de 4 ans) qui utilise la détection de couleurs au lieu du codage traditionnel. Les enfants programment Indi en plaçant des tuiles en silicone colorées sur le sol – le robot roule dessus et réagit à chaque couleur par un comportement spécifique. Cette approche intuitive sans écran rend la programmation accessible aux enfants qui ne savent pas encore lire tout en enseignant les concepts de la pensée computationnelle.

Indi propose une logique de programmation unique « SI couleur ALORS action » : SI le robot voit du vert, ALORS il avance ; SI il voit du violet, ALORS il tourne à gauche. Cette pensée conditionnelle est fondamentale dans toute programmation.

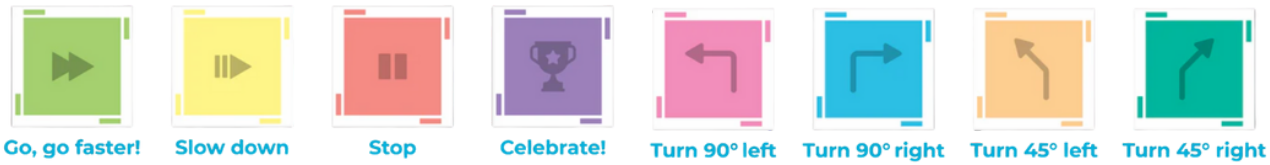
Le robot

Indi est un robot compact en forme de voiture avec un capteur de couleur intégré sur sa face inférieure. Le capteur lit la couleur des tuiles placées sous le robot et déclenche le comportement correspondant. Caractéristiques principales :

- Capteur de couleur intégré pour détecter les tuiles
- LED qui changent de couleur pour correspondre aux couleurs détectées
- Batterie rechargeable (charge USB-C)
- Conception robuste pour les jeunes apprenants
- Calibrage automatique de la rotation

Les tuiles de couleur

Indi est livré avec un ensemble de tuiles en silicone colorées. Chaque couleur déclenche un comportement par défaut spécifique :



Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Prise en main

Étape 1 : Découvrir comment Indi réagit à chaque tuile de couleur.
Étape 2 : Construire des labyrinthes avec des combinaisons infinies de tuiles colorées.
Étape 3 : Renforcer les compétences de résolution de problèmes en naviguant dans les puzzles complexes que vous créez.

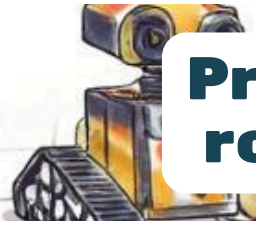
Créer vos propres couleurs

Certains souhaitent créer leurs propres tuiles, cartes ou chemins pour Indi. Gardez à l'esprit que chaque couleur est programmée dans le firmware et instantanément reconnaissable par Indi. D'autres sources de couleur, comme le papier de construction ou les feutres, peuvent ne pas offrir une expérience aussi cohérente que les tuiles et/ou cartes de couleur fournies avec votre Indi.

Le site officiel d'Indi fournit toutes les couleurs Pantone en libre accès : https://sphero.com/pages/sphero-indi. N'hésitez pas à créer autant de chemins que possible et amusez-vous !

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Tutoriel et programmation



Programmer un robot Micro:bit

Introduction

Le BBC micro:bit est une carte de programmation polyvalente qui peut être transformée en cerveau de nombreux projets robotiques. Contrairement à Cubetto ou Sphero Indi, qui sont des systèmes autonomes, les robots micro:bit sont construits à partir de composants séparés : la carte micro:bit elle-même, une carte d'extension (aussi appelée « shield » ou « breakout board »), des moteurs, des capteurs et un châssis ou système de construction.

Cette modularité apporte une grande flexibilité mais introduit également un concept important : chaque plateforme robotique nécessite d'être programmée sur MakeCode en utilisant des extensions spécifiques.

Qu'est-ce que MakeCode ?

MakeCode est un environnement de programmation gratuit, basé sur navigateur, développé par Microsoft. Il propose une approche de codage visuelle par blocs qui rend la programmation accessible aux débutants tout en offrant des alternatives textuelles pour les utilisateurs plus avancés. Caractéristiques principales :

- Gratuit et en ligne : Aucune installation requise ; fonctionne dans tout navigateur web moderne
- Utilisable hors ligne : Une application téléchargeable est disponible pour Windows, macOS et Linux
- Multi-langages : Supporte les blocs, JavaScript et Python
- Simulateur inclus : Tester les programmes sans matériel physique
- Documentation complète : Tutoriels, exemples et ressources enseignants intégrés

Accéder à MakeCode : https://makecode.microbit.org/

MakeCode propose trois façons d'écrire le même programme :

- Mode blocs (par défaut) : Programmation visuelle utilisant des blocs colorés et imbriqués. Idéal pour les débutants et les jeunes élèves. Chaque bloc représente un concept de programmation sans nécessiter la mémorisation de syntaxe.
- Mode JavaScript : Programmation textuelle utilisant la syntaxe JavaScript. Les blocs se convertissent automatiquement en JavaScript, et vice versa. Utile pour les apprenants passant au codage textuel ou pour écrire une logique plus complexe.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

- Mode Python : Programmation textuelle utilisant la syntaxe Python. Disponible dans les versions récentes de MakeCode. Particulièrement précieux car Python est largement enseigné dans l'enseignement secondaire et utilisé professionnellement.

Les trois modes sont interconnectés – les modifications dans un mode se reflètent dans les autres (avec quelques limitations pour le code Python complexe). Cela permet une transition progressive de la programmation visuelle vers la programmation textuelle.

Pour les élèves de 8 à 12 ans, nous recommandons d'utiliser le mode blocs car il semblera aux enfants très logique et naturel de donner des instructions à leur robot.

Si vous utilisez une carte micro:bit pour la première fois

- Ouvrir MakeCode dans votre navigateur.
- Connecter la carte à l'ordinateur : Utiliser un câble micro-USB. La carte apparaîtra comme une clé USB nommée « MICROBIT ». Votre carte sera automatiquement associée à votre éditeur sur MakeCode. Sinon, vous pouvez l'associer manuellement en cliquant sur le bouton « Connecter l'appareil » dans le menu accessible en cliquant sur « ... » à côté du bouton « Télécharger ».
- Transférer le programme : Une fois votre code créé, cliquez sur « Télécharger » et le programme se téléchargera automatiquement sur la carte associée. Sinon, vous pouvez copier le fichier .hex sur la carte. La LED orange clignote pendant le transfert.
- Alimentation autonome : Ajouter un pack de piles AAA dans le support fourni s'il n'est pas inclus dans votre kit robotique.

Pour plus d'informations : micro:bit - Guide de démarrage - Introduction

L'écosystème des extensions

Les blocs MakeCode intégrés à la carte micro:bit couvrent les fonctions de base : affichage LED, boutons, accéléromètre, boussole, radio et entrée/sortie simple sur les broches. **Cependant, les plateformes robotiques utilisent du matériel spécialisé qui nécessite des blocs de programmation personnalisés**. Chaque fabricant crée une extension MakeCode qui fournit :

- Des blocs dédiés pour leurs moteurs et pilotes de moteurs spécifiques
- Des blocs de capteurs calibrés pour leur matériel
- Des commandes simplifiées qui masquent le code de bas niveau complexe
- Des blocs visuels correspondant à la terminologie de leur produit

Un programme écrit avec l'extension d'un robot ne fonctionnera pas avec un robot différent, même si les deux utilisent une carte micro:bit. L'extension doit correspondre au matériel.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Construire un robot avec PowerBricks

Un robot PowerBricks typique est construit à base de briques techniques LEGO et utilise deux moteurs DC (M1 et M2) entraînant les roues gauche et droite pour la direction différentielle. Vous pouvez ajouter un capteur ultrasonique à l'avant pour la détection d'obstacles, mais ce n'est pas nécessaire pour ce projet.

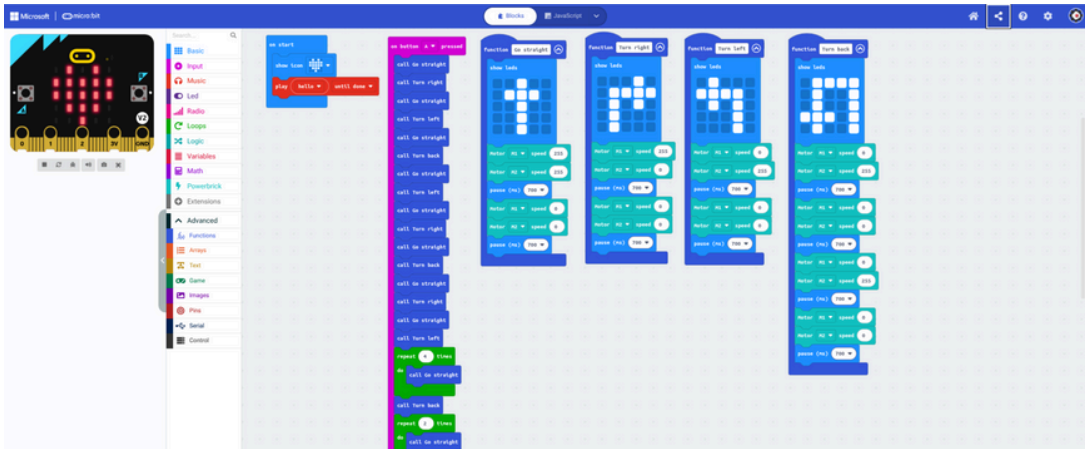
La direction différentielle contrôle le robot en faisant varier la vitesse de chaque roue :

Moteur gauche (M1)	Moteur droit (M2)	Résultat
Avant	Avant	Tout droit
Arrière	Arrière	Marche arrière
Stop	Avant	Pivot à gauche
Avant	Stop	Pivot à droite
Arrière	Avant	Rotation sur place à gauche
Avant	Arrière	Rotation sur place à droite

Les valeurs de vitesse vont de −255 (pleine marche arrière) à 255 (pleine marche avant). Zéro arrête le moteur.

Programme de navigation sur grille : un exemple de classe

Le programme suivant montre un robot naviguant sur un chemin prédéfini sur une grille. Il a été développé pour être utilisé avec le curriculum d'Apprentissage Bio-inspiré, où les élèves mémorisent d'abord un itinéraire eux-mêmes, puis programment le robot pour suivre le même chemin. Cet exemple réel de classe illustre plusieurs concepts clés de programmation : les fonctions, l'exécution séquentielle, les boucles, le retour visuel et le calibrage du timing.
Le programme est visible ici : https://makecode.microbit.org/S25194-48798-31124-89886



Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Dans cette séquence, le robot doit aller de « Départ » à l'arrivée visible sur la feuille de mémorisation exemple en case B6. En rouge, vous pouvez voir les informations du modèle créé par l'élève, et en bleu, les flèches décrivent le chemin que l'élève veut faire prendre au robot avec cet objectif : depuis le départ, atteindre l'arrivée en traversant le plus de cases possible.

Structure du programme

Le programme est organisé en sections distinctes :

1. Code de démarrage : S'exécute une fois lorsque la carte micro:bit est mise sous tension
2. Définitions des fonctions (Tourner_gauche, Tourner_droite, Aller_tout_droit, Faire_demi_tour) : Primitives de mouvement réutilisables
3. Gestionnaire d'événement (input.onButtonPressed) : La séquence de navigation principale déclenchée par le bouton A

Cette structure suit les bonnes pratiques de programmation : définir d'abord ses outils, puis les utiliser dans le programme principal.

Organiser le code avec des fonctions

Plutôt que de disperser les commandes moteur dans tout le programme, nous définissons quatre fonctions de mouvement : Aller_tout_droit, Tourner_gauche, Tourner_droite et Faire_demi_tour. Chaque fonction encapsule une action complète – afficher une flèche directionnelle sur les LED, faire tourner les moteurs appropriés pendant une durée calibrée, puis s'arrêter et faire une pause avant le mouvement suivant.

Cette approche transforme le programme principal en quelque chose de lisible. Au lieu de déchiffrer powerbrick.MotorRun(M1, 255) suivi de powerbrick.MotorRun(M2, 255), on lit simplement Aller_tout_droit(). Le code devient une narration du parcours du robot que les élèves peuvent suivre sans comprendre le contrôle moteur sous-jacent.

Les fonctions simplifient également la maintenance. Si le robot dépasse systématiquement les cases de la grille, on ajuste le timing à un seul endroit plutôt que de chercher dans des dizaines de valeurs de pause dispersées. Pour le débogage, on peut tester chaque fonction isolément avant de les combiner en séquences complexes.

Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium:

Boucles pour un code plus propre

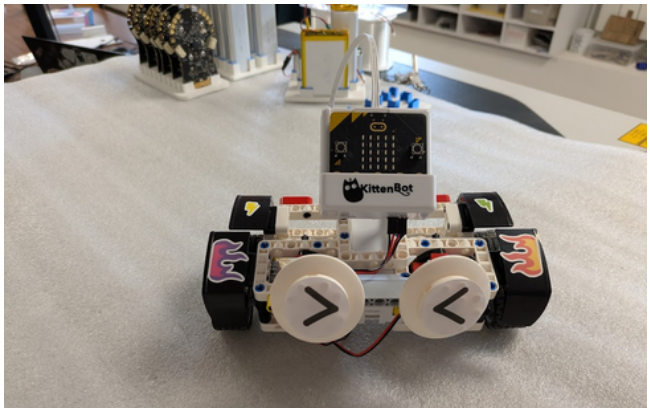
Lorsque le robot doit parcourir quatre cases d'affilée, on pourrait écrire Aller_tout_droit quatre fois. À la place, une boucle répète l'appel de fonction, rendant l'intention parfaitement claire : « avancer quatre fois ». Au-delà de la propreté, les boucles réduisent les erreurs – il n'y a aucun risque de copier accidentellement trois fois au lieu de quatre – et elles introduisent le concept fondamental d'itération que les élèves rencontreront tout au long de leur parcours de programmation.

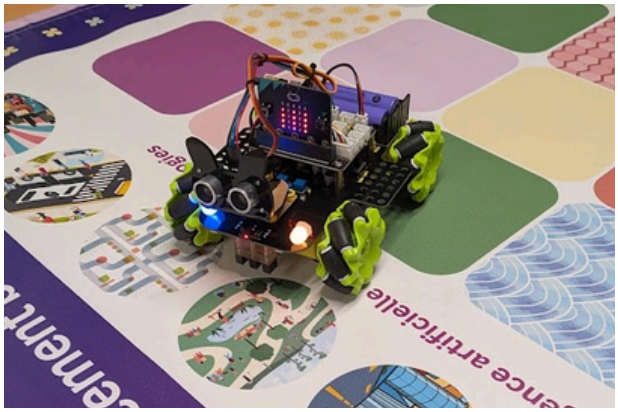
Retour visuel et calibrage

Chaque mouvement affiche une flèche LED personnalisée correspondant à la direction, permettant aux enseignants et aux élèves de « lire » les intentions du robot. Si l'affichage montre une flèche à gauche mais que le robot tourne à droite, le diagnostic est immédiat : les connexions des moteurs sont inversées.

Flux du programme

Au démarrage, une icône cœur et un son d'accueil confirment que le robot est prêt et en attente. Appuyer sur le bouton A lance la séquence de navigation. À la fin, un motif en damier et un son de célébration apportent une conclusion. Cette structure claire – prêt, en cours d'exécution, terminé – aide les élèves à comprendre les états du programme et fournit un retour satisfaisant pour leur travail.





Numéro de projet : 2023-1-FR01-KA220-SCH-000151881
Cofinancé par l'Union européenne
Consortium: