



MAILAGE NUMÉRIQUE AU SERVICE DE LA
CRÉATIVITÉ DANS LES ENSEIGNEMENTS STEAM

guide d'utilisation de la brique logicielle **magentics**

Remerciements

Le projet magnetics est une initiative portée par le Laboratoire d'Aix-périmentation et de Bidouille (fablab d'Aix-en-Provence).

Financé par le dispositif edu-up, magnetics a duré 2 ans, entre Novembre 2020 et Décembre 2022 avec pour objectif de construire une brique logicielle permettant de créer un maillage numérique dédié à la programmation à l'école. Au travers de magnetics, l'équipe du L.A.B souhaite participer à la promotion des outils numériques et de la programmation dans les classes au service d'une pédagogie active et innovante dédiée à l'enseignement interdisciplinaire et créatif des STEAM (Sciences, Technologies, Ingénierie, Arts, Mathématiques).

Le L.A.B remercie particulièrement le Ministère de l'Éducation Nationale et de la Jeunesse pour son soutien. Au travers du dispositif edu-up, le ministère soutient la production de solutions numériques innovantes et adaptées. Ces solutions contribuent au maintien de la continuité pédagogique dans le respect de la liberté pédagogique de chacun. Elles prennent en compte les besoins de tous les élèves et répondent aux exigences de l'école inclusive. Elles sont mises à disposition des enseignants et de leurs élèves le plus souvent en accès direct et gratuit ou bien après inscription libre et volontaire, à tout ou partie de la ressource.

En savoir plus sur le dispositif edu-up :
<https://eduscol.education.fr/1603/le-dispositif-edu-up>





Crédits et contributions

Contributeurs

Le L.A.B remercie tous les contributeurs au projet magnetics qui ont apporté leur savoir-faire dans la construction de ce projet : Sébastien Nedjar (coordinateur), Manon Ballester (gestionnaire), Jonathan Baudin (Développeur), Pascal Flores (Développeur), Adrien Virieux (Développeur).

Le L.A.B remercie également tous les membres du comité scientifique et pédagogique permanent mais également les acteurs du monde de l'éducation et de la recherche qui sont intervenus plus ponctuellement pour nous aider à construire ce projet : Mercé Gisbert, Georgios Mavromanolakis, Cindy Smits, Roberto Canónico, Carme Grimalt, Mickaël Martin-Nevot, Margarida Romero, Patricia Corieri, Emmanuel Feller, Yannick Marietti et Stéphane Vassort.

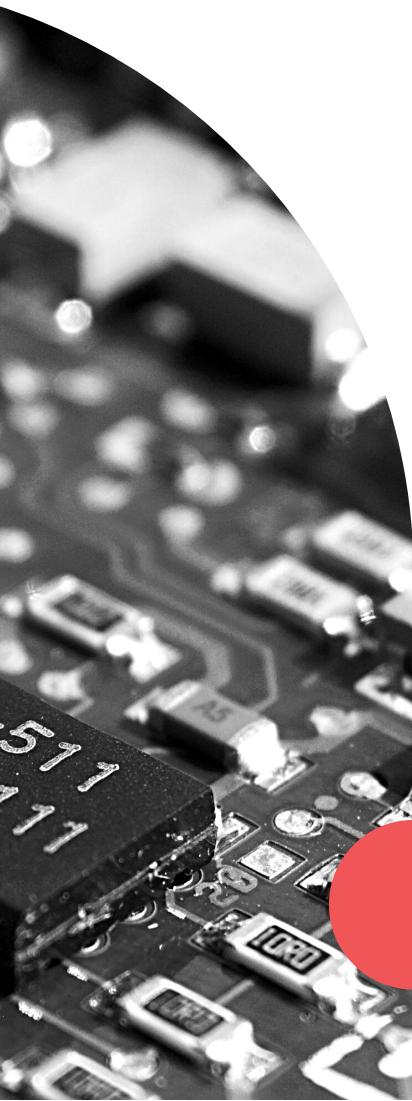
Crédits

Captures d'écran réalisées par les auteurs principalement sur makecode.lets-steam.eu et python.let-steam.eu

Icônes réalisées par Freepik à partir de www.flaticon.com

Licence et droits

Réalisé avec le soutien du Ministère de l'Education Nationale
L'ensemble des productions réalisées dans le cadre du projet magnetics est soumis à une licence Creative Commons Attribution-ShareAlike 4.0 International, qui permet une utilisation, distribution et reproduction sans restriction sur n'importe quel support, à condition de citer de manière appropriée l'auteur ou les auteurs originaux et la source, de fournir un lien vers la licence Creative Commons ainsi que d'indiquer si des modifications ont été apportées et de les partager de la même manière.



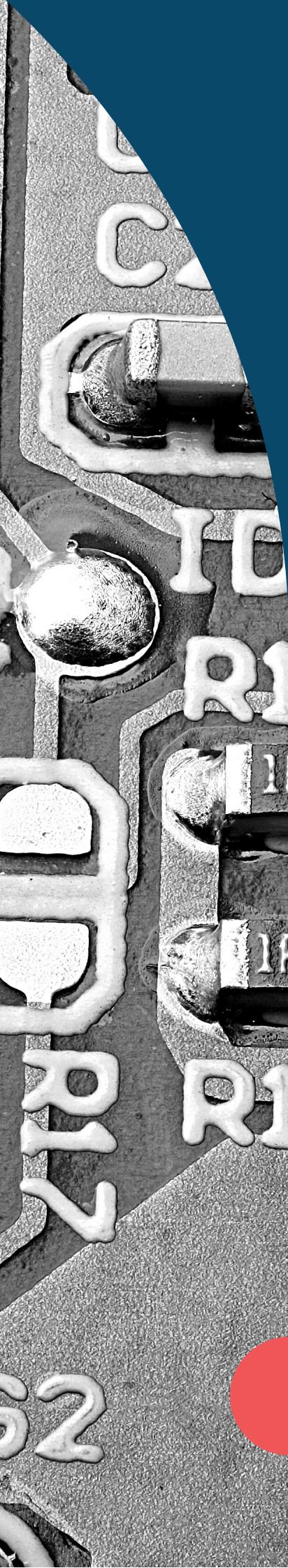


Table des matières

page 5

Propos introductifs

Le projet magnetics

Introduction aux modalités d'apprentissage par enquête

page 11

Les outils autour de magnetics – Guide d'utilisation

Les plateformes

Les cartes programmables

page 40

Programmer la brique magnetics

Sur MakeCode

Sur Micropython

Sur MakeCode pour Micro:bit

page 62

S'inspirer – Idées de projet

page 72

Aller plus loin – Découvrir les technologies derrière magnetics

Introduction

Le projet magnetics participe à la promotion de l'utilisation d'outils numériques et de la programmation au service de la mise en œuvre d'expérimentations stimulantes et créatives dans le domaine de l'enseignement STEAM (Sciences, Technologie, Ingénierie, Arts et Mathématiques). Il vise à renforcer l'intégration de la programmation au sein des classes, en donnant accès aux enseignants et aux élèves à des outils de collecte de données adaptés aux besoins éducatifs.

Grâce à la diffusion des ressources magnetics, adossées à d'autres initiatives promouvant la programmation à l'école, le projet souhaite participer au développement des compétences des enseignants et des élèves dans le domaine de l'informatique et de la production numérique.

Le projet magnetics prend principalement la forme d'une brique technique logicielle, implantée dans les plateformes d'apprentissage de la programmation les plus populaires : Scratch, MakeCode et CircuitPython.

Elle a pour fonctionnalité de permettre à l'utilisateur de construire des projets multicartes, afin de collecter un ensemble plus large de données sur un terrain d'expérimentation étendu, sans avoir à connecter l'ensemble des cartes entre elles.

Par exemple, une carte unique peut jouer le rôle de collecteur de données tandis que les autres cartes ne sont qu'émettrices, et communiquent leurs informations sans connaissance préalable sur la typologie du réseau ni même sur le routage sous-jacent.

Ce développement est basé sur l'utilisation de la technologie de réseau maillé Bluetooth Low Energy Mesh (BLE Mesh) compatible avec toutes les cartes programmables disposant d'un module Bluetooth Low Energy. D'un point de vue pédagogique, l'accès à la brique magnetics permet aux enseignants qui l'utilisent de pouvoir lancer des expérimentations nouvelles et de plus grande envergure tout en gardant un environnement logiciel et matériel connu et maîtrisé.

Concrètement, le projet magnetics permet d'atteindre les objectifs suivants :

- Promouvoir la pédagogie active, la créativité et la curiosité pour les sciences en collectant et traitant des données nouvelles auxquelles les élèves n'auraient pas accès sans l'utilisation des cartes programmables et des capteurs, via la mise en place d'expérimentation de large échelle.
- Développer les compétences des enseignants et des élèves dans le domaine de l'informatique et de la production numérique.
- Mieux aborder les objets connectés à l'école en incitant à intégrer de nouveaux outils numériques au service des apprentissages.

Le prototype magnetics est aujourd'hui disponible gratuitement, couvert par la licence ouverte Creative Commons BY SA. Il a été implémenté dans les plateformes MakeCode (<https://makecode.lets-steam.eu/>) et Micropython (<https://python.lets-steam.eu/>) et l'ensemble des codes sources sont disponibles pour tous sur GitHub (<https://github.com/letssteam>).

Ce guide a pour objectif de donner aux enseignants toutes les informations utiles à la compréhension de ce qu'est magnetics, de son usage éducatif et de l'utilisation qui peut en être fait dans la classe.

Ainsi, ce manuel comprend à la fois des ressources pratiques pour appréhender les outils de programmation, les cartes, des fiches d'activité pour programmer des cartes électroniques en utilisant la brique magnetics, des idées de projet pour vous inspirer dans le lancement de vos projets de programmation mais également des ressources techniques pour les plus technophiles d'entre vous ou simplement pour aiguiser votre curiosité ! Bonne lecture !



Introduction aux modalités d'apprentissage par enquête

Le projet magnetics a pour vocation de participer à l'émergence au sein de la classe de dispositifs d'apprentissage par enquête et basés sur l'expérimentation. Afin de comprendre et réutiliser l'approche IBL (Inquiry-Based Learning ou apprentissage basée sur l'expérimentation ou l'enquête), nous proposons d'utiliser les supports de formation du projet Let's STEAM, conçus pour comprendre l'intérêt de l'IBL et d'aider les enseignants à créer des expérimentations complexes à résoudre au sein de la classe. Le projet magnetics est une brique additionnelle permettant d'enlever une contrainte technique et matérielle et ainsi de créer des projets de plus larges échelles. Couplé à des outils tels que les communications LoRa (notamment développés par Vittascience pour la carte STM32), magnetics propose de s'affranchir des limites d'utilisation de plusieurs cartes, afin de construire des enquêtes plus ambitieuses.

L'IBL est une stratégie éducative flexible comprenant des phases le souvent organisées en cycles et divisées en sous-phases avec des connexions logiques en fonction du contexte d'investigation. Cette méthode comporte cinq phases générales (orientation, conceptualisation, investigation, conclusion et discussion) et sept sous-phases (questionnement, génération d'hypothèses, exploration, expérimentation, interprétation des données, réflexion et communication), qui sont présentées ici.



La méthode IBL peut être utilisée afin de conceptualiser une manière structurée de mettre en œuvre des activités de recherche et de développer des projets éducatifs multidisciplinaires dans les salles de classe. L'IBL n'est pas une procédure linéaire et les apprenants doivent être impliqués dans diverses formes d'expérimentation, en passant par différentes combinaisons de phases, mais pas nécessairement toutes. Par exemple, si l'analyse des données n'est pas suffisamment satisfaisante, les élèves peuvent revenir à la phase de conceptualisation et reconsidérer leur question et/ou leur plan expérimental. Lorsque les élèves arrivent à une conclusion, de nouvelles questions peuvent être générées, et le processus recommence de manière progressive. La description des processus de l'approche IBL par Pedaste et al. comprend les cinq phases décrites ci-dessous :

Orientation

L'orientation est la phase où se produit l'identification du problème. Le sujet à étudier est présenté et l'intérêt pour une situation problématique qui peut être résolue par une expérimentation est stimulé. Le sujet à étudier doit être en rapport avec la vie quotidienne, les intérêts et les connaissances préalables des élèves. Le rôle de l'enseignant dans cette phase est d'encourager les élèves à exprimer leurs idées, leurs connaissances préalables et leurs questions sur le sujet tout en favorisant l'interaction et la communication entre eux. Par exemple, les élèves peuvent créer des cartes conceptuelles de ce qu'ils savent, ne savent pas ou veulent savoir sur le sujet étudié. Ce type d'activités peut également être utile pour les phases suivantes de l'expérimentation.

Conceptualisation

La phase de conceptualisation fait référence à la compréhension du concept, qui se rapporte à la situation problématique présentée au moment de l'orientation. Elle se divise en deux sous-phases (questionnement et génération d'hypothèses) qui conduisent l'apprenant à la phase d'investigation. Le rôle de l'enseignant est d'aider les élèves à comprendre comment ils peuvent formuler des questions et/ou des hypothèses qui peuvent mener à une investigation. Si les élèves ne sont pas familiarisés avec les sous-phases de questionnement et de génération d'hypothèses, l'enseignant peut choisir un type d'enquête structuré dans un premier temps, puis évoluer vers des types d'enquête plus ouverts afin de fournir les conseils appropriés.

Investigation

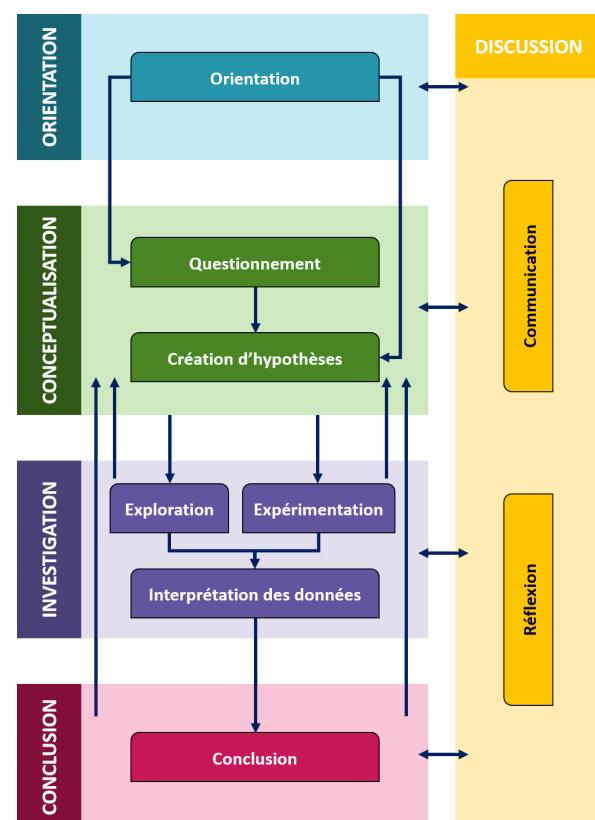
L'investigation est la phase au cours de laquelle les élèves collectent des preuves afin de répondre à leurs questions et/ou de tester leur hypothèse. Elle comprend les sous-phases d'exploration, d'expérimentation et d'interprétation des données. L'enseignant fournit les ressources dont les élèves peuvent avoir besoin et les maintient sur la bonne voie afin que le processus qu'ils choisissent de suivre permette de répondre à la question d'investigation. Les élèves doivent déterminer et rassembler ce qui constitue une preuve. S'ils ne sont pas familiers avec ce processus, ils peuvent choisir un type d'expérimentation structuré. L'enseignant peut fournir ou encourager les élèves à créer des moyens (par exemple, des tableaux, des graphiques, etc.) qui peuvent les aider à organiser, classer et analyser les données.

Conclusion

Dans cette phase, les élèves tirent des conclusions sur la base de la question d'investigation et de l'interprétation des données. Le rôle de l'enseignant durant cette phase, une comparaison entre les données interprétées, les prédictions et hypothèses initiales (que les élèves ont exprimées durant la phase d'orientation), peut être stimulée. Ce processus peut également déboucher sur de nouvelles hypothèses et questions sur le sujet étudié.

Discussion

Au cours de la phase de discussion, les élèves articulent leurs résultats en les communiquant aux autres et/ou en réfléchissant à tout ou partie des étapes d'expérimentation pendant le processus et/ou à la fin de celui-ci. Le rôle de l'enseignant est d'encourager la collaboration afin que les élèves puissent présenter leurs résultats et leurs idées, fournir des arguments et donner un retour aux autres. S'ils ne sont pas familiers avec ces pratiques, l'enseignant peut leur fournir des lignes directrices qui les aideront à communiquer pendant toutes les phases de l'expérimentation.



Types d'expérimentation

Les types d'expérimentation varient afin que les élèves soient activement impliqués dans le processus dans la mesure où ils·elles sont compétent·e·s et capables de le faire. Le type d'expérimentation qu'un enseignant peut choisir de suivre dépend fortement des objectifs de la leçon, de l'âge des élèves, de leur participation antérieure à l'expérimentation et des compétences scientifiques qu'ils·elles ont déjà acquises. Comme indiqué ci-dessous, plus l'élève est responsable, moins il·elle reçoit de directives et plus l'expérimentation est ouverte. Les variations des types d'expérimentation concernent donc l'implication croissante ou décroissante de l'enseignant et de l'élève dans le processus. L'expérimentation structurée est dirigée par l'enseignant afin que les élèves atteignent un résultat spécifique, tandis que dans l'expérimentation mixte, les élèves sont plus impliqués au cours de l'investigation, les conseils de l'enseignant restant prédominants. Ces formes d'expérimentation sont généralement choisies lorsque les élèves sont initiés aux pratiques d'observation et lorsqu'il s'agit de développer une compétence ou un concept spécifique. L'expérimentation ouverte offre davantage de possibilités de développer des compétences scientifiques, étant donné que les élèves travaillent directement avec les ressources et les pratiques, d'une façon qui ressemble à une approche scientifique authentique. Par exemple, si les élèves n'ont pas d'expérience préalable de la conception d'investigations et de la collecte de données, il convient de choisir une forme d'expérimentation plus structurée et/ou guidée. Lorsque les élèves ont acquis les compétences nécessaires, ils peuvent passer à des activités de recherche plus ouvertes. À un moment donné, les élèves devraient participer à toutes les formes d'expérimentation, tout en passant progressivement d'une forme d'expérimentation à une autre, avec une progression simultanée de la complexité et de l'autonomie.

partie 1

Les outils autour de magentics - Guide d'utilisation

sous-partie 1 - les éditeurs



MakeCode

À propos du codage par blocs et de Microsoft MakeCode

En 1975, Seymour Papert, du MIT Media Lab, a créé un langage de programmation pour débutants appelé LOGO. Il l'a développé sur la base de recherches qui ont montré que jouer avec des blocs de code était un moyen particulièrement efficace d'enseigner les concepts de programmation.

Papert a inventé le terme "constructionnisme" pour décrire la manière dont les apprenants construisent de nouvelles connaissances en s'appuyant sur des connaissances établies.

Les blocs de MakeCode sont eux-mêmes des modèles de la manière dont un nouvel apprentissage se produit par l'application de concepts dans un environnement d'apprentissage ouvert.

Les langages de programmation basés sur des blocs tels que Scratch et MakeCode s'appuient sur les recherches de Papert et constituent un excellent moyen pour les élèves de commencer à apprendre les concepts de codage sans avoir à se soucier des questions de syntaxe et des problèmes techniques.

Recherche

ENTRÉE

BROCHES

BOUCLES

LOGIQUE

VARIABLES

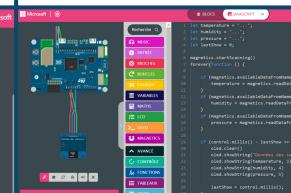
MATHS

AVANCÉ

Présentation de MakeCode

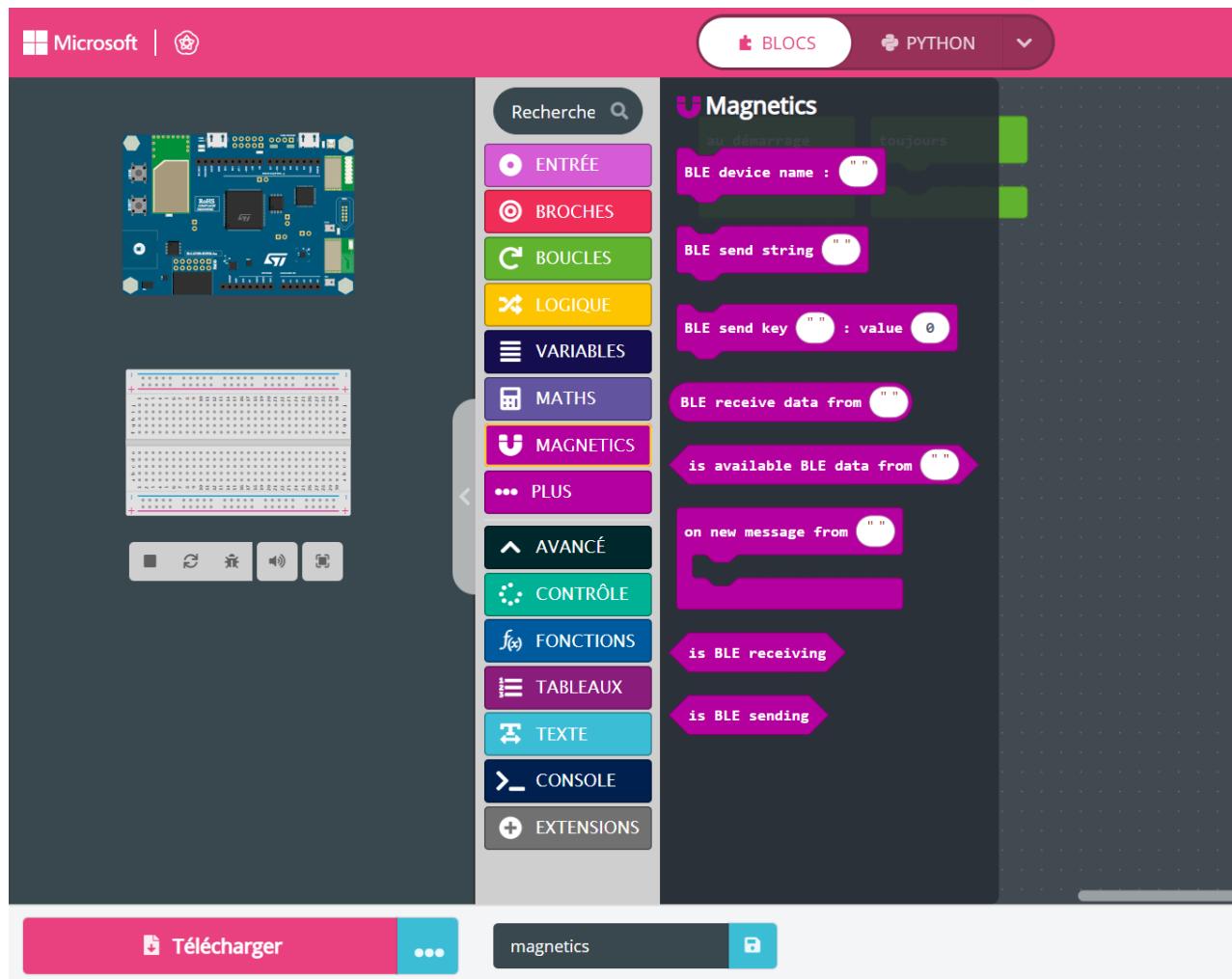
Microsoft MakeCode est une plateforme gratuite et open source permettant de créer des expériences d'apprentissage de l'informatique qui favorisent la progression vers la programmation dans le monde réel. Elle donne vie à l'informatique pour tous les élèves avec des projets amusants, des résultats immédiats et des éditeurs de blocs et de texte pour les apprenants de différents niveaux. Microsoft MakeCode est un projet conjoint entre Microsoft Research et Visual Studio qui vise à simplifier la programmation de dispositifs à base de microcontrôleurs à l'aide d'une application Web moderne. L'outil permet aux utilisateurs de découvrir la programmation en utilisant des "blocs" pour représenter les instructions et les structures de contrôle. Il est également possible de convertir les blocs en langage JavaScript ou Python (vice-versa), ce qui permet de découvrir et d'apprendre progressivement la programmation. Enfin, un simulateur permet de tester le code sans avoir recours à une carte électronique physique (elle-même, ainsi que les capteurs/actionneurs, leur comportement et les connexions, sont simulés).

MakeCode est basé sur les outils suivants :

SIMULATEUR	ÉDITEUR PAR BLOC	ÉDITEUR JAVASCRIPT
		

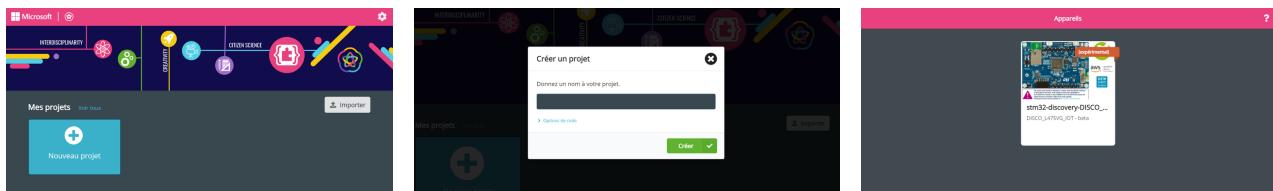
L'éditeur MakeCode offre un écosystème pertinent de par son accessibilité, son interface, la possibilité de programmer en 3 langages différents (donc adaptable à plusieurs niveaux du début du collège au lycée), sa compatibilité multicarte, sa capacité à fournir une simulation visuelle des activités du capteur (intéressant notamment dans le cas d'un enseignement à distance, ou d'un manque de ressources pour équiper les élèves de cartes individuelles), sa compatibilité RGPD, et surtout, la possibilité de développer des projets simples à complexes, pouvant ou non communiquer avec le terrain.

Les développements Magnetics ont donc été directement intégrés à un éditeur MakeCode spécifique développé lors du projet Let's STEAM et accessible ici : <https://makecode.lets-steam.eu/>. L'utilisation de cet éditeur particulier avait pour ambition d'intégrer le projet magnetics à une ressource conçue pour l'enseignement, comprenant d'autres fonctionnalités pertinentes.



Commencer à utiliser MakeCode

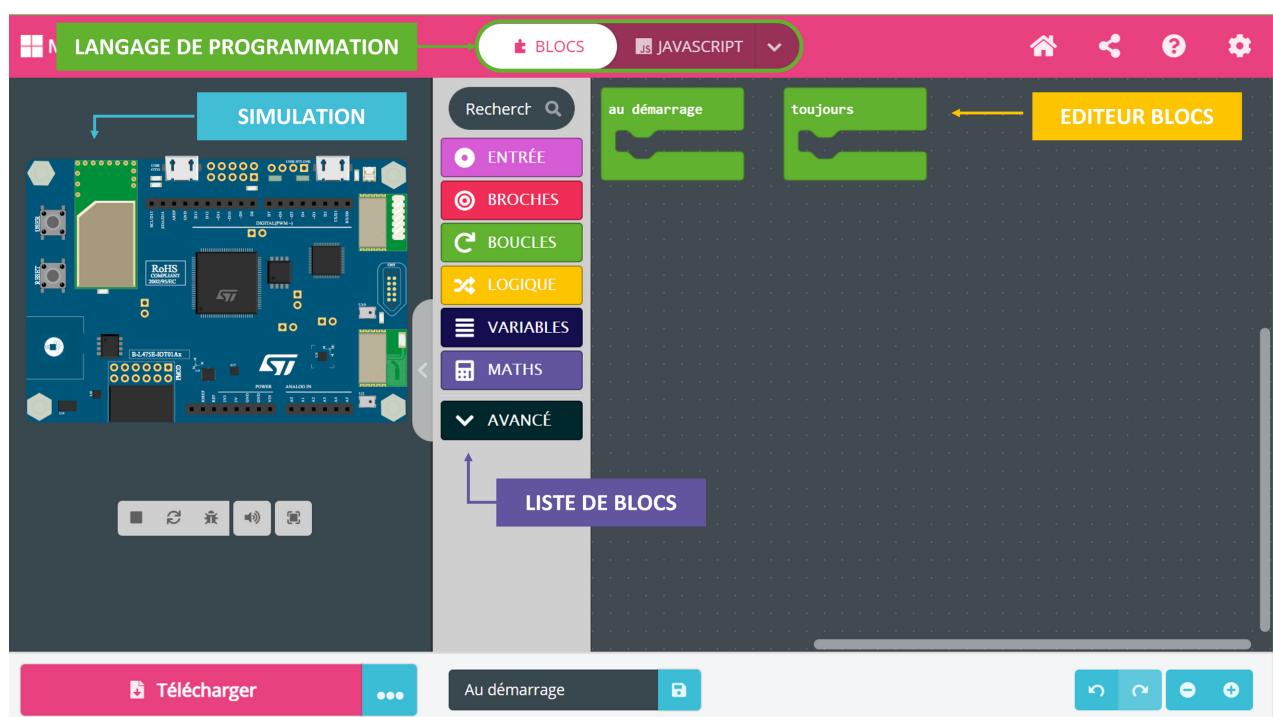
Lorsque vous entrez sur l'interface MakeCode de Let's STEAM, vous arrivez directement sur la page d'accueil. Sur cette page, vous pouvez créer un nouveau projet, ouvrir un projet existant si vous avez déjà travaillé sur l'éditeur, voir les cartes supportées et découvrir des ressources inspirantes. Lorsque vous créez un projet, il est important de le nommer avec un titre clair et compréhensible, vous permettant d'annoncer clairement l'objectif du programme. Sur l'écran suivant, vous devrez choisir la carte sur laquelle vous allez travailler. Sur les fiches d'activités de Let's STEAM et magnetics, tous les exemples ont été développés à l'aide de la carte STM32 IoT Node.



Si l'interface chargée est affichée en anglais au lancement, vous pouvez changer la langue en cliquant sur le bouton "paramètre" afin de voir les versions supportées.



Une fois la carte sélectionnée, vous aurez alors accès à l'éditeur, présenté ci-dessous :



Voici les composants de l'éditeur :

Le SIMULATEUR (à gauche de l'éditeur)

Un simulateur interactif fournit aux élèves un retour immédiat sur comment leur programme fonctionne et leur permet de tester et de déboguer leur code.

La LISTE DES BLOCS au centre

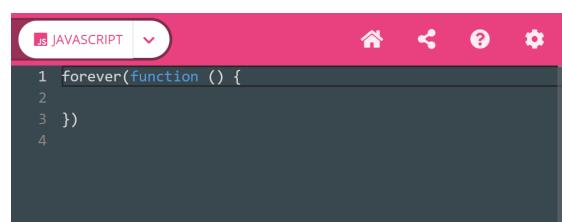
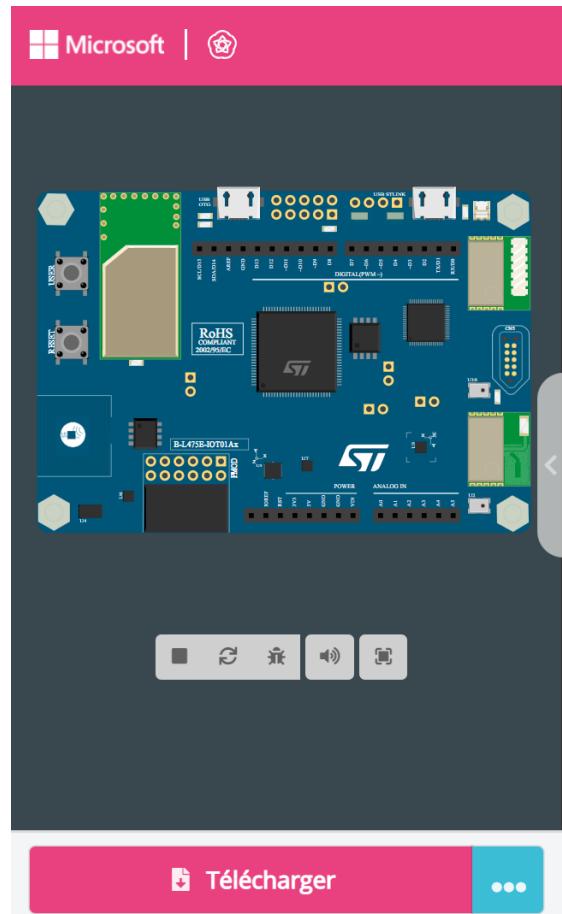
pouvant être utilisés dans votre programme, ainsi qu'un champ de recherche.

L'ÉDITEUR DE BLOCS sur la partie droite

qui comprend déjà deux fonctions communes à toutes les activités : "on start" et "forever loop". Les élèves qui débutent dans le programmation peuvent commencer par des blocs de couleur qu'ils peuvent glisser et déposer sur leur espace de travail pour construire leurs programmes.

Dans l'éditeur, vous pourrez également choisir le mode de programmation :

- Par le biais de blocs (voir la fiche d'activité RIASI - Faire clignoter une LED)
- Par l'intermédiaire d'un éditeur JavaScript
- Par le langage Python pour les élèves plus avancés.



Voici la liste des blocs de base disponibles sur l'éditeur MakeCode de Let's STEAM. Vous aurez un aperçu plus précis de la fonction de chaque bloc dans les diverses fiches d'activités proposées dans ce manuel :

Entrée		ENTRÉE	Utiliser un capteur dans votre programme (comme un bouton, un thermomètre, etc.).
Broches		BROCHES	Interagir directement avec les broches (fréquemment appelées pin) et modifier leur état
Control		CONTROLE	Gérer l'exécution des événements
Boucles		BOUCLES	Mettre en œuvre les répétitions
Logique		LOGIQUE	Effectuer des tests, des comparaisons et des opérations de logique booléenne.
Variables		VARIABLES	Créer des variables et des compteurs
Maths		MATHS	Effectuer divers calculs mathématiques
Fonctions		FONCTIONS	Créer des sous-programmes
Tableaux		TABLEAUX	Créer une valeur ou un texte dans un tableau
Texte		TEXTE	Modifier les textes
Console		CONSOLE	Afficher les données
Extensions		EXTENSIONS	Accéder à la liste des extensions disponibles dans la version de MakeCode
Magnetics		MAGNETICS	Gérer les communications
Datalogger		DATALOGGER	Créez un jeu de données pour enregistrer les données des capteurs
LCD		LCD	Afficher du texte ou des informations sur un écran (LCD)
OLED		OLED	Afficher du texte ou des informations sur un écran (OLED)
Music		MUSIC	Extension pour jouer de la musique



CircuitPython

Ladyada et Adafruit – Pionniers dans le monde de l'open source hardware

En 2005, Limor "Ladyada" Fried, ingénierie au MIT, a fondé Adafruit Industries pour faire découvrir le monde de l'électronique à tous ceux qui veulent apprendre.

L'entreprise est spécialisée dans la vente et la production de composants électroniques et de matériel libre et vise à diffuser des ressources pour apprendre l'électronique, les sciences et l'ingénierie, plus particulières auprès des femmes, des enfants et des novices.

En 2009, Limor reçoit le Pioneer Award par l'Electronic Frontier Foundation pour sa participation à la communauté de l'open source hardware et software. Nommée en 2011 parmi les Femmes les plus influentes en technologie par le magazine Fast Company, elle est devenue la première femme ingénierie en vedette sur la couverture de Wired.

Dans une interview accordée à CNET, Ladyada résume sa démarche « S'il y a une chose que j'aimerais voir, ce serait que des enfants se disent "je peux faire ça" et qu'ils commencent ainsi leur parcours pour devenir ingénieurs et entrepreneurs ».

Limor Fried a été nommée "Entrepreneur de l'Année" en 2012 par le magazine Entrepreneur, sur les 15 finalistes, elle était la seule femme.



Présentation de CircuitPython

CircuitPython est un projet open source dérivé de Micropython qui utilise le langage de programmation Python pour programmer des cartes à base de microcontrôleurs. Il est conçu pour simplifier l'expérimentation et l'apprentissage du code sur des cartes à microcontrôleurs à bas prix. Avec CircuitPython, aucun téléchargement initial n'est nécessaire. Les principaux atouts de la plate-forme CircuitPython sont les suivants :

Rapide et facile

Créez un fichier, modifiez votre code, enregistrez le fichier et il s'exécute immédiatement. Il n'y a pas besoin de compiler, de télécharger ou de mettre en ligne.

Facile à utiliser pour les débutants

CircuitPython a été conçu dans un souci d'éducation. Il est facile de commencer à apprendre à coder et vous obtenez un retour immédiat du tableau.

Mises à jour faciles du code

Puisque votre code vit sur le disque dur, vous pouvez le modifier quand vous le souhaitez, vous pouvez également conserver plusieurs fichiers pour faciliter l'expérimentation.

Console série + REPL

Ils permettent un retour en direct de votre code et une programmation interactive.

Stockage des fichiers

Le stockage interne de CircuitPython le rend idéal pour l'enregistrement de données, la lecture de clips audio et l'interaction avec des fichiers.

Strong Hardware Support

Il existe de nombreuses bibliothèques et pilotes pour les capteurs, les cartes d'extension et autres composants externes.

Python est le langage de programmation qui connaît la croissance la plus rapide. Il est enseigné dans les écoles et les universités car c'est un langage de programmation de haut niveau, ce qui signifie qu'il est conçu pour être plus facile à lire, à écrire et à maintenir. Il prend en charge les modules et les paquets. Il possède un interpréteur intégré, ce qui signifie qu'il n'y a pas d'étapes supplémentaires, comme la compilation, pour que votre code fonctionne.

Commencer à utiliser Python grâce à notre éditeur Micropython

Pour magnetics, nous utiliserons un éditeur en ligne, offrant une expérience similaire à MakeCode en termes de facilité d'utilisation et de compatibilité avec les navigateurs prenant en charge la technologie Web-USB.

Sur cet éditeur, les développements spécifiques réalisés lors de Let's STEAM permettent de proposer une expérience utilisateur beaucoup plus adaptée à une utilisation avec des étudiants.

Par exemple, auparavant, il était nécessaire de changer le port USB et les cavaliers chaque fois que l'on voulait utiliser MicroPython, alors que maintenant tout peut être fait par le même port USB que celui utilisé par MakeCode.

Aussi la reprogrammation partielle du programme permet d'attendre moins longtemps pour tester un programme qui a été modifié par un étudiant et ainsi avoir une boucle écriture/simulation/déploiement/correction quasi instantanée.

Pour bien supporter la carte, les pilotes des capteurs des fiches d'activités ont été ajoutés.

Cet outil est disponible ici : <http://python.lets-steam.eu/>.

L'éditeur Python a été conçu en pensant particulièrement aux débutants en matière de codage en mode texte. L'interface claire et colorée et la gamme de fonctionnalités visent à réduire les obstacles à la prise en main et sont conçues pour attirer un large éventail d'étudiants, aidant ainsi davantage de jeunes à s'orienter vers un avenir numérique optimal.

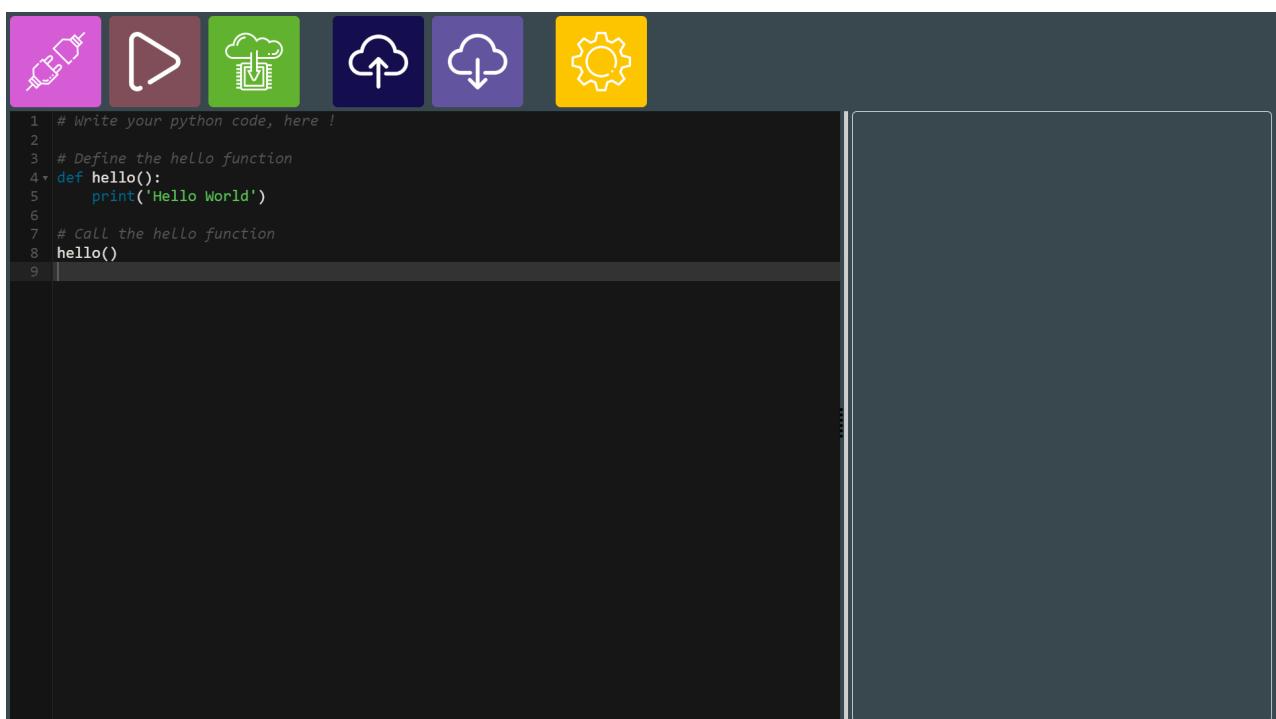
L'éditeur est divisé en deux panneaux :

Menu latéral gauche

Par défaut, il affiche la section Référence. Utilisez les boutons sur le côté gauche pour basculer entre les menus Référence, Idées, API, Projet, Paramètres et Aide.

Fenêtre d'édition

Il s'agit de votre code Python. Tapez ici et tapez pour apporter des modifications ou faites glisser et déposez du code depuis le menu latéral gauche dans cette zone.



The screenshot shows a dark-themed Python code editor. At the top, there is a horizontal toolbar with six colored icons: a pink square with a plug icon, a maroon triangle, a green cloud with a gear, a dark blue cloud with an upward arrow, a purple cloud with a downward arrow, and a yellow gear. Below the toolbar, a code editor window displays the following Python code:

```
1 # Write your python code, here !
2
3 # Define the hello function
4 def hello():
5     print('Hello World')
6
7 # Call the hello function
8 hello()
9
```



Scratch

Scratch Foundation – Permettre à tous les enfants d'accéder à la programmation

Scratch a été lancé en mai 2007 sous la forme d'une application téléchargeable. Les enfants pouvaient créer leurs propres histoires interactives, jeux et animations sur leur ordinateur et les partager avec la communauté en ligne en les téléchargeant sur le site Web de Scratch.

La sortie de Scratch 2.0 a porté Scratch sur le web en 2013, entraînant une croissance exponentielle de la communauté. Scratch est désormais la plus grande plateforme de codage pour enfants au monde. La communauté ayant besoin d'un moyen de soutenir la croissance continue de Scratch et d'aider le projet à poursuivre son évolution, la Fondation Scratch a été créée en 2013 en tant qu'organisation indépendante à but non lucratif.

Mitchel Resnick, professeur de recherche sur l'apprentissage au Media Lab du MIT, et David Siegel, cofondateur et coprésident de la société de gestion des investissements Two Sigma, sont les fondateurs initiaux.

Mitch et David partagent la même vision : faire en sorte que Scratch soit disponible gratuitement, pour que tous les enfants du monde puissent exprimer leur créativité à travers le code. Depuis 2014, la Fondation Scratch a fourni près de 5 millions de dollars de financement.



Présentation de Scratch

Scratch est un langage de programmation ouvert et libre associé à une communauté en ligne, conçu et maintenu par le groupe Lifelong Kindergarten au MIT Media Lab. En utilisant Scratch, les jeunes apprennent à penser de façon créative, à collaborer, et à raisonner suivant une logique système.

Avec Scratch, les élèves peuvent programmer leurs propres histoires interactives, jeux et animations - et partager leurs créations avec d'autres membres de la communauté en ligne. Scratch aide les jeunes à apprendre à penser de manière créative, à raisonner systématiquement et à travailler en collaboration - des compétences essentielles pour la vie au XXI^e siècle. Scratch est conçu spécialement pour les jeunes de 8 à 16 ans.

Scratch est utilisé dans plus de 150 pays différents et est disponible dans plus de 40 langues. Les élèves apprennent avec Scratch à tous les niveaux (de l'école primaire au collège) et dans toutes les disciplines (comme les mathématiques, l'informatique, les langues, les arts, les études sociales). De mai 2009 à mai 2019, plus de 27 000 éducateurs qui soutiennent l'apprentissage avec le langage de programmation Scratch ont partagé 4 749 messages de discussion, 1 027 ressources et 354 histoires avec la communauté en ligne.

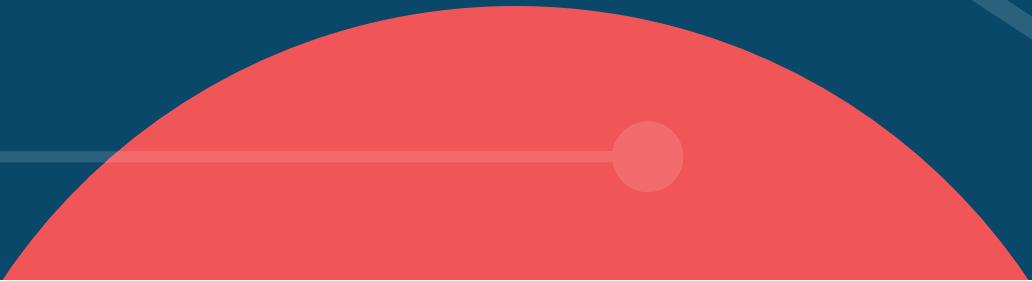
Le projet Scratch a bénéficié du soutien financier de la National Science Foundation, de la Scratch Foundation, de la Siegel Family Endowment, de Google, de la LEGO Foundation, d'Intel, de Cartoon Network, de la Lemann Foundation et de la MacArthur Foundation. Voir la page des crédits pour plus d'informations.

Scratch est orienté multimédia avec pour objectif l'enseignement de l'univers informatique aux enfants ou aux débutants, il est basé sur la manipulation des objets, des sons et vidéos. L'application permet de modifier le code du programme en cours d'exécution. Scratch fonctionne grâce à un éditeur visuel, tout le code est directement inscrit dans la langue maternelle de l'utilisateur (plus de quarante langues sont disponibles) sous forme de briques en couleurs. Il permet de mettre en œuvre visuellement des concepts de base de la programmation tels que les boucles, les conditions, les tests, les affectations de variables. Scratch permet ainsi à l'enseignant de diffuser sa pédagogie au moyen de l'interactivité dans une approche ludique.

partie 1

Les outils autour de magnetics - Guide d'utilisation

sous-partie 2 – les cartes



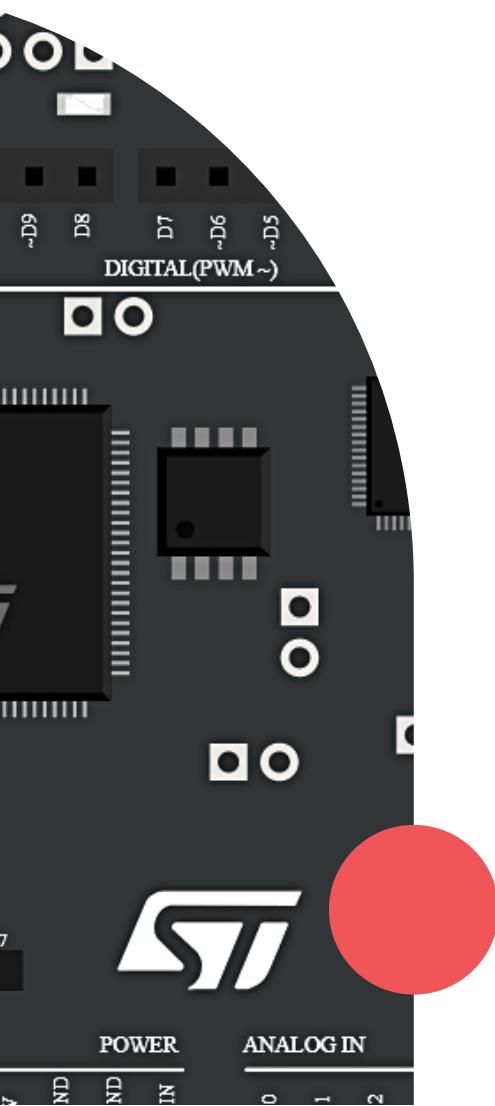
STM32 Education

Collaboration pour l'éducation

Fruit d'une collaboration entre l'académie d'Aix-Marseille et la société STMicroelectronics (Roussset), la carte STM32Education permet de donner du sens à la programmation et à l'algorithmie en proposant une carte complète grâce à laquelle les élèves vont pouvoir concevoir des objets connectés et les programmer simplement.

Cette carte est destinée à de nombreuses disciplines par la richesse de ses capteurs embarqués. La carte STM32Educ est de conception française, son adoption est soutenue par des financements pour l'éducation.

Au-delà de la carte Micro:Bit, elle propose plus de fonctionnalités et un set de capteurs permettant d'intégrer des aspects plus riches aux projets scientifiques et interdisciplinaires qui seront ciblés par Magnetics.



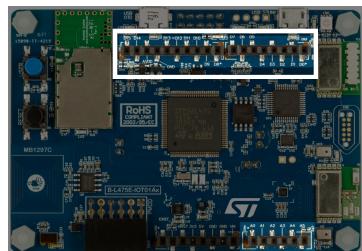
Découvrir la carte STM32 IoT Node et son ensemble de capteurs

La carte "STM32 IoT Node" est une carte programmable, ce qui signifie qu'elle est capable d'exécuter des programmes créés par l'utilisateur.

Pour exécuter ce programme, la carte dispose d'un "microcontrôleur", qui est en quelque sorte son cerveau (voir ci-contre). Par exemple, Le nom du microcontrôleur de notre carte présentée ici est : STM32L475VG.

Les GPIO

Comme nous pouvons le constater, il y a beaucoup de "pattes" ou de "broches" sur la carte, appelées "General Purpose Input / Output" (ou GPIO en abrégé). Il est possible de les utiliser pour interagir avec des éléments externes. Même s'il y a beaucoup de GPIO, il n'est pas possible de tous les utiliser. Les GPIO utilisables sont situés en haut et en bas de la carte. Les blocs noirs percés sont appelés "blocs de broches". En regardant attentivement, nous pouvons remarquer les noms des GPIO inscrits autour (par exemple en bas à droite : "D0, D1, D2, D3, ..., A0, A1, A2, ...").



Nous découvrirons les différences entre les broches Ax (A0, A1, ...) et Dx (D0, D1, D2, ...) plus loin dans les activités.

Il reste un autre bloc de broches, celui-ci est spécial, c'est un "power pinout block". Nous pouvons utiliser ces broches pour alimenter des capteurs ou des actionneurs (comme un moteur, une lumière, etc.).



L'inscription sur le dessus du bloc de broches informe sur la manière de l'utiliser. Le "5V" correspond au "+" (pôle positif) d'une batterie et le "GND" (abréviation de "Ground") au "-" (pôle négatif).

Les périphériques

La différence entre le nombre de GPIO disponibles via le bloc de broches et le nombre de pattes du microcontrôleur s'explique par la présence de multiples périphériques déjà connectés au microcontrôleur, disponibles sur la carte "STM32 IoT Node" elle-même. La présence de tous ces périphériques rend cette carte particulièrement accessible, car elle permet de mettre en œuvre un large éventail d'activités, des plus simples aux plus complexes, et des plus basiques aux plus ludiques. C'est un véritable atout pour réaliser des activités entraînantes en classe.

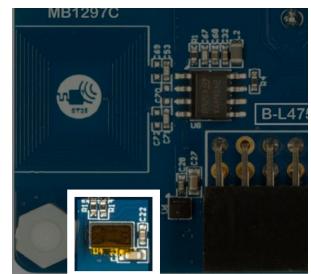
Boutons

Sur le côté gauche de la carte, vous trouverez deux boutons. Le bouton noir est le bouton RESET, permettant au programme de redémarrer si nécessaire. L'autre (bleu) peut être utilisé dans un programme pour détecter quand l'utilisateur appuie sur ce bouton (appui court, appui long, relâchement, etc.). Il peut être utile pour créer des interactions simples avec l'utilisateur, comme un buzzer dans le cadre d'organisation de concours à l'aide de cette carte.



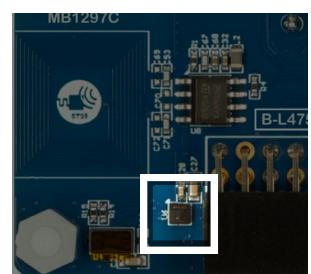
Capteur de distance

Dans le coin inférieur gauche de la carte, juste à droite de la vis en nylon, vous pouvez trouver un capteur pour mesurer la distance. Il est officiellement appelé "temps de vol" (time of flight) parce qu'il mesure le temps que met un rayon laser à faire des allers-retours (voler) entre le capteur et un objet.



Capteur de température et d'humidité

Sur la droite du capteur "temps de vol", on trouve un capteur à la fois thermomètre et hygromètre ("2 en 1"). Cela peut être utile pour mettre en œuvre des activités liées à la surveillance de la chaleur ou pour aborder des notions de météorologie.



Capteur accéléromètre et gyroscope

Juste au-dessus du "power pinout block", se trouve un capteur à la fois accéléromètre et gyroscope ("2 en 1"). L'accéléromètre est utilisé pour mesurer l'accélération. Vous pouvez l'utiliser pour détecter les mouvements de la carte (par exemple, si la carte est secouée). Le gyroscope donne des informations sur l'inclinaison de la carte. Ce capteur fonctionne sur trois axes (X, Y et Z), ce qui implique qu'il est possible de détecter les mouvements dans l'espace 3D.



Capteur de pression atmosphérique

À côté du capteur accéléromètre et gyroscope, vous trouverez un petit capteur appelé baromètre. Ce capteur nous donne la valeur de la pression atmosphérique.



Capteur magnétométrique

À côté du capteur de pression atmosphérique, vous pouvez voir le magnétomètre. Il est utilisé pour récupérer la valeur d'un champ magnétique. Il peut également mesurer des valeurs sur trois axes (X, Y et Z).



Microphone

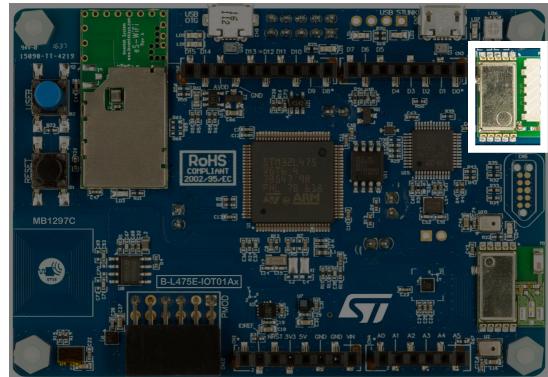
Dans le coin en bas à droite, vous pouvez voir le microphone, utile pour capturer des sons.



Les modules

Module Bluetooth

En haut à droite de la carte, vous pouvez trouver le module bluetooth. Il peut être utilisé pour communiquer et échanger des données avec d'autres appareils (comme une autre carte STM32 IoT Node, ou votre téléphone).



Connecteurs Micro-USB

En haut de la carte, vous pouvez voir deux connecteurs micro-USB. Le port USB de droite est celui que vous utiliserez le plus souvent, car il permet de connecter la carte à votre ordinateur et de transférer le programme que vous aurez fait sur MakeCode au microcontrôleur. Le port de gauche, appelé "port USB OTG", permet de programmer la carte pour qu'elle agisse et soit reconnue comme un autre dispositif tel qu'un clavier, une souris ou une manette de jeu.



STeaM32 et connecteurs Jacdac

Une nouvelle carte pour mieux répondre aux besoins des enseignants

Fruit d'une coopération d'envergure dans le cadre du projet I-NOVMICRO porté par le Campus d'Excellence Industrie du Futur – GIP FCIP et financé dans le cadre du PIA3, la carte pédagogique STeaM32 est un support matériel pour les activités d'apprentissage de la programmation dans l'enseignement secondaire.

Outre les aspects purement techniques et technologiques importants pour illustrer les concepts, la carte est adaptée dans sa forme et son utilisation pour être utilisée dans une classe avec des élèves de 10 à 18 ans.

Une grande partie de sa facilité de mise en œuvre provient de son intégration logicielle dans les différentes plateformes éducatives telles que Makecode, Scratch et CircuitPython.

Cependant, la forme et la facilité d'utilisation sont d'une grande importance pour obtenir un produit qui répond aux besoins des enseignants.

Découvrir la carte STeAM32 et son ensemble de capteurs

La carte STeAM32 est basé sur la famille STM32 de microcontrôleurs 32-bits en circuits intégrés réalisés par la société Franco-Italienne STMicroelectronics. Le processeur d'application STM32WB55 est le lieu d'exécution des programmes utilisateurs. Une seule application complète, comprenant le code utilisateur, le code d'exécution et la pile Bluetooth, est chargée et exécutée directement depuis la mémoire flash de la puce. Toutes les broches GPIO accessibles à l'utilisateur sont fournies par ce processeur. Il y a un moteur radio 2.4GHz intégré utilisé pour fournir des capacités Bluetooth via une antenne hors puce.

Communication sans fil Bluetooth

Le transceiver 2.4GHz embarqué supporte les communications Bluetooth via le MCU M0, qui fournit une pile Bluetooth Low Energy entièrement qualifiée. Cela permet à la carte de communiquer avec une large gamme de périphériques Bluetooth, y compris les smartphones et les tablettes.

Boutons

Les six boutons à l'avant de la carte, et le bouton 1 à l'arrière, sont des boutons tact momentanés de type "push to make". Le bouton arrière est connecté au processeur STM32WB55 pour la réinitialisation du système. Cela signifie que l'application se réinitialisera indépendamment du fait qu'elle soit alimentée par USB ou par batterie. Les boutons avant peuvent être programmés dans l'application utilisateur pour n'importe quel usage. Ils sont débités par le logiciel, ce qui inclut également la détection des pressions courtes et longues. Les boutons fonctionnent dans un mode électrique inversé typique, où une résistance de rappel garantit un '1' logique lorsque le bouton est relâché, et un '0' logique lorsque le bouton est enfoncé. Les boutons A et B sont connectés aux broches GPIO qui sont également accessibles sur le connecteur de bord.

Écran

L'écran est un écran circulaire IPS TFT couleur pour du texte et des graphiques haute résolution. Il est connecté au processeur d'application. Le logiciel d'exécution rafraîchit de façon répétée cet écran à une vitesse élevée, de sorte qu'il se trouve dans la plage de persistance de la vision de l'utilisateur et qu'aucun scintillement ne soit détecté.

Capteur de mouvement

La carte est équipée d'une puce combinant un accéléromètre et un magnétomètre qui fournit une détection sur 3 axes et une détection de l'intensité du champ magnétique. Elle comprend également une détection matérielle de certains gestes (comme la détection de chute) et une détection de gestes supplémentaires (par exemple, logo vers le haut, logo vers le bas, secousse) via des algorithmes logiciels. Un algorithme logiciel dans le runtime standard utilise l'accéléromètre embarqué pour transformer les lectures en une lecture de boussole indépendante de l'orientation de la carte. La boussole doit être étalonnée avant d'être utilisée, et le processus d'étalonnage est automatiquement lancé par le logiciel d'exécution. Ce dispositif est connecté au processeur d'application via le bus I2C.

Broches d'entrée/sortie à usage général

Le connecteur de bord fait ressortir de nombreux circuits GPIO du processeur d'application. Certains de ces circuits sont partagés avec d'autres fonctions de la carte, mais beaucoup de ces circuits supplémentaires peuvent être réaffectés à un usage général si certaines fonctions logicielles sont désactivées.

Alimentation électrique

L'alimentation de la carte peut se faire via la connexion USB, via la puce d'interface (qui possède un régulateur intégré), ou via une batterie branchée sur le connecteur supérieur. Il est également possible (avec précaution) d'alimenter la carte à partir du pad 3V situé en bas. Le pad 3V du bas peut être utilisé pour fournir une petite quantité de puissance aux circuits externes.

Interface

La puce gère la connexion USB, et est utilisée pour flasher un nouveau code sur la carte, envoyer et recevoir des données série vers votre ordinateur principal.

Communications USB

La carte possède une pile de communication USB, qui est intégrée dans le firmware du bootloader. Cette pile permet de glisser et déposer des fichiers sur le disque de stockage de masse afin de charger du code dans le processeur d'application. Elle permet également de transmettre des données en série vers et depuis le processeur d'application via USB vers un ordinateur hôte externe, et supporte le protocole CMSIS-DAP pour le débogage hôte des programmes d'application.

Débogage

Le processeur peut être utilisé avec des outils hôtes spéciaux pour déboguer le code qui s'exécute sur le processeur d'application. Il se connecte au processeur d'application via 4 fils de signal. Le code du processeur peut également être débogué via son interface de débogage logiciel SWD intégrée, par exemple pour charger le code initial du chargeur de démarrage dans ce processeur au moment de la fabrication, ou pour récupérer un chargeur de démarrage perdu.

L'intégration de connecteurs Jacdac

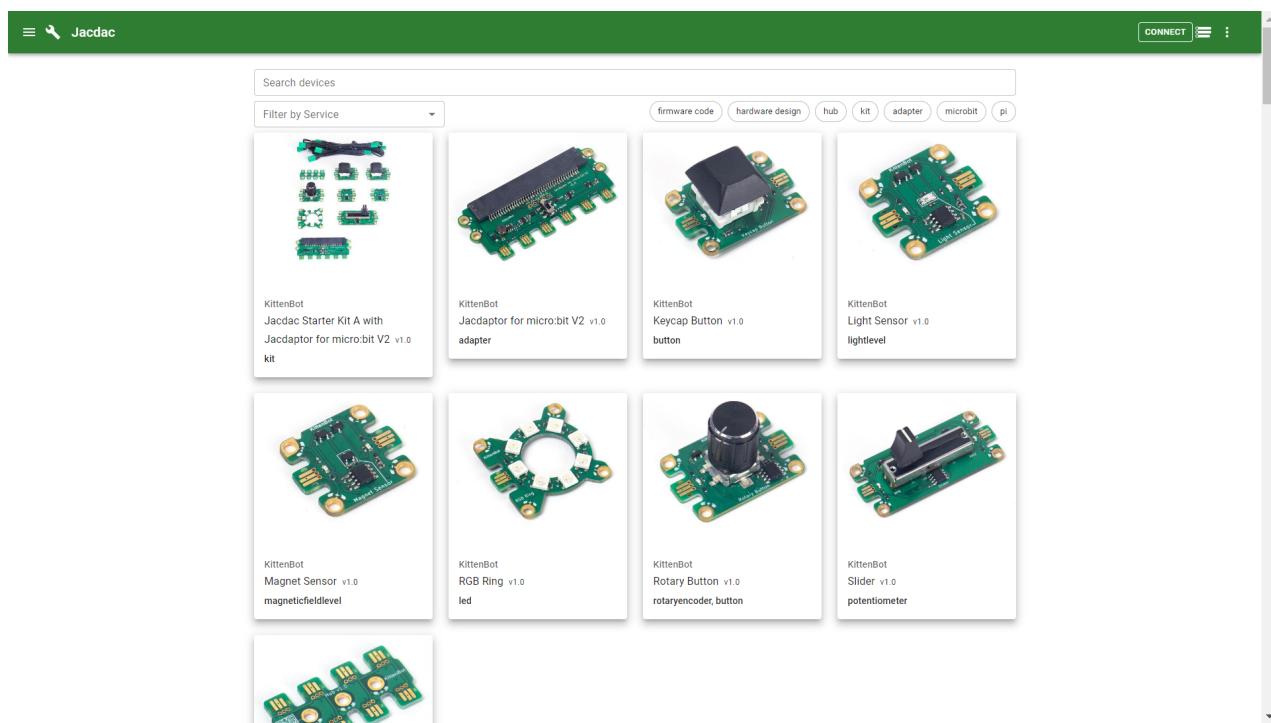
En complément des modules et éléments présentés ci-dessus, la carte STEaM32 intègre deux connecteurs Jacdac ainsi qu'un connecteur Micro:bit permettant de connecter un hub Jacdac afin de câbler un nombre plus important de capteurs. L'environnement Jacdac est un système de connectique plug and play pour ajouter en cascade ses capteurs/actionneurs à la volée. Une fois connectés, les périphériques sont reconnus automatiquement ainsi que les services associés, ce qui permet d'ajouter des capteurs à tout instant.

L'écosystème Jacdac a été conçu pour être convivial et adapté à l'éducation. Les dispositifs Jacdac communiquent à l'aide de paquets sur un bus, où chaque dispositif s'annonce et présente son ensemble de services. Chaque dispositif Jacdac possède un minuscule microcontrôleur qui exécute le protocole Jacdac et communique sur le bus. Les paquets Jacdac sont envoyés en série entre les dispositifs physiques sur le bus Jacdac et peuvent également être envoyés sur WebUSB/WebBLE, fournissant une connectivité à des outils et services basés sur le Web et fonctionnant dans un navigateur Web.

Dans le cadre du magnetics, le Jacdac fournit donc une solution parfaite pour assurer que :

- La solution reste bon marché et accessible pour les enseignants tout en offrant plus de fonctionnalités que les tableaux actuels. En effet, le Jacdac peut être ajouté à un circuit imprimé pour quelques centimes.
- La solution est flexible, basée sur un processus "plug-and-play" avec des outils web étendus.
- La solution est extensible afin d'être adaptée aux services et aux dispositifs nécessaires/utilisés par les enseignants.

Vous pouvez découvrir le catalogue de capteurs et autres composants open source développés dans le cadre du projet Jacdac et évaluer le potentiel de modularité offert par l'intégration de cette technologie à la carte STeAM32 (<https://microsoft.github.io/jacdac-docs/devices/>) :





micro:bit

La carte éducation de la BBC

Micro:bit est une carte programmable miniature (4cm x 5cm) facile à mettre en œuvre.

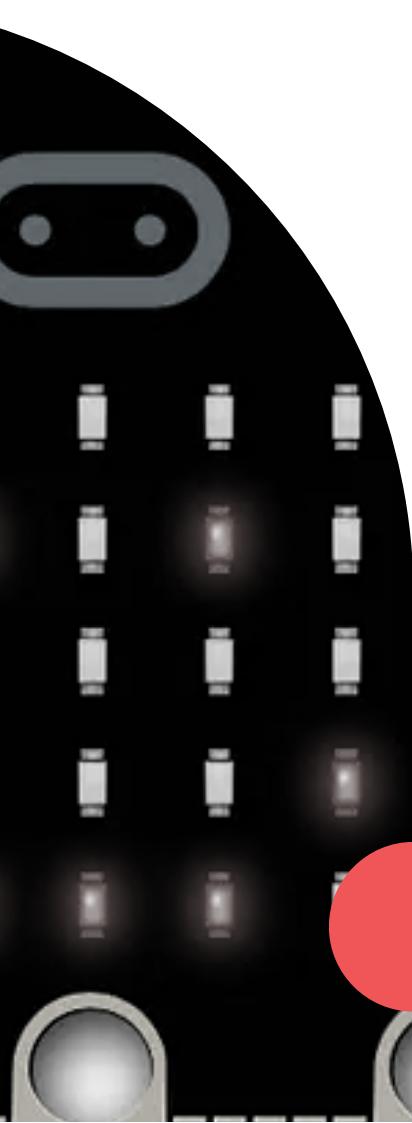
Elle permet de découvrir la programmation d'objets réels sans connaître rien ni de l'électronique, ni de concepts informatiques.

Cette carte a été conçue par la BBC pour permettre aux plus jeunes de s'initier au code. Elle dispose pour cela de de nombreuses entrées-sorties programmables et d'une connexion Bluetooth.

Elle est comparable aux cartes Arduino dont elle se distingue par une facilité d'emploi améliorée.

Aujourd'hui Micro:bit est l'une des cartes les plus populaires et utilisées par le monde éducatif.

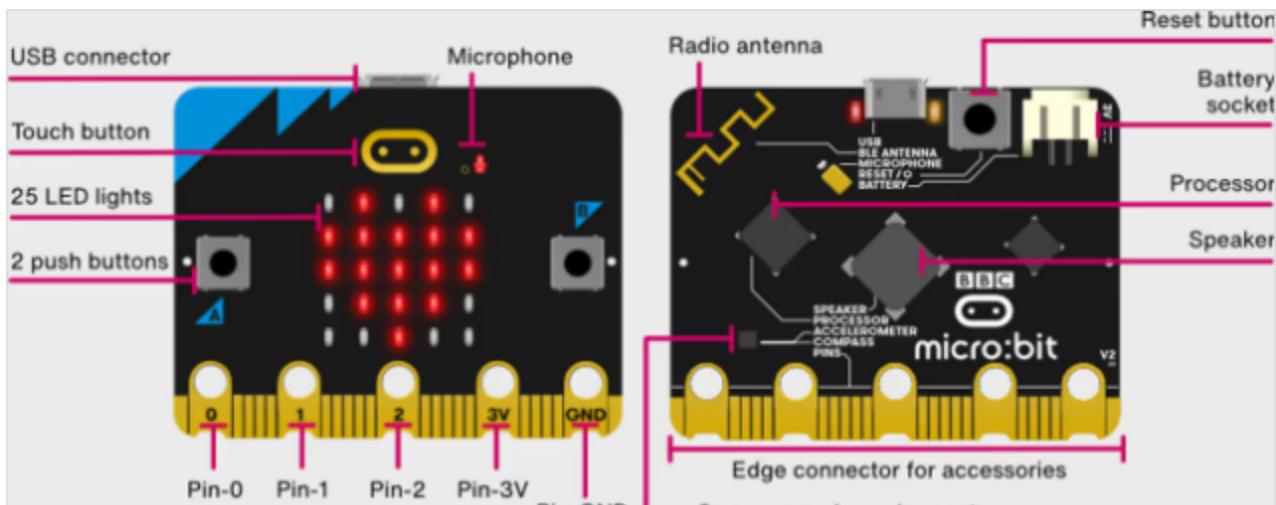
La carte Micro:bit est actuellement utilisée autant avec Scratch, Makecode et CircuitPython. Son intégration poussée au sein de ces outils en font une carte de choix pour tous les professeurs.



Découvrir le guide de l'utilisateur par micro:bit

Vous pouvez trouver sur le site de micro:bit un guide d'utilisation très complet vous permettant de comprendre toutes les fonctionnalités offertes par la carte. Nous vous proposons d'explorer cette ressource et de vous en servir comme référence.

Vous pouvez trouver le guide ici : <https://microbit.org/fr/get-started/user-guide/overview/>



Aperçu du BBC micro:bit

En savoir plus sur les fonctionnalités de votre micro:bit

 [microbit_edu](#)



Série de microcontrôleurs ESP32

Une carte appréciée dans l'univers de l'IoT

ESP32 est une série de microcontrôleurs de type système sur une puce (SoC) d'Espressif Systems, basé sur l'architecture Xtensa LX6 de Tensilica, intégrant la gestion du Wi-Fi et du Bluetooth (jusqu'à LE 5.0 et 5.11) en mode double, et un DSP.

Le principal outil de développement est ESP-IDF, logiciel libre développé par Espressif, écrit en C et utilisant le système temps réel FreeRTOS.

Il intègre un nombre important de bibliothèques et on retrouve dans son écosystème des bibliothèques tierces libres pour différents types de périphériques liés à l'embarqué et au temps réel.



Découvrir la carte ESP32

L'ESP32 peut fonctionner comme un système autonome complet ou comme un dispositif esclave d'un MCU hôte, ce qui réduit la surcharge de la pile de communication sur le processeur d'application principal. L'ESP32 peut s'interfacer avec d'autres systèmes pour fournir des fonctionnalités Wi-Fi et Bluetooth via ses interfaces SPI / SDIO ou I2C / UART. Conçu pour les appareils mobiles, l'électronique portable et les applications IoT, l'ESP32 atteint une consommation d'énergie ultra-faible grâce à l'association de plusieurs types de logiciels propriétaires. L'ESP32 comprend également des fonctionnalités de pointe, telles que le gating d'horloge à grain fin, divers modes d'alimentation et la mise à l'échelle dynamique de la consommation. L'ESP32 est capable de fonctionner de manière fiable dans des environnements industriels, avec une température de fonctionnement allant de -40°C à +125°C. Grâce à des circuits d'étalonnage avancés, l'ESP32 peut éliminer dynamiquement les imperfections des circuits externes et s'adapter aux changements des conditions extérieures.

Caractéristiques

Wi-Fi et Bluetooth Dual Mode

L'intégration des technologies Wi-Fi, Bluetooth et Bluetooth LE permet de cibler un large éventail d'applications et de rendre nos modules réellement polyvalents. L'utilisation du Wi-Fi assure la connectivité dans un large rayon, tandis que l'utilisation du Bluetooth permet à l'utilisateur de détecter facilement un module (avec des balises à faible énergie) et de le connecter à un smartphone.

Haute intégration

Avec des commutateurs d'antenne, des baluns RF, des amplificateurs de puissance, des amplificateurs de réception à faible bruit, des filtres et des modules de gestion de l'alimentation intégrés, nos puces ajoutent une fonctionnalité et une polyvalence inestimables à vos applications avec des exigences minimales en matière de circuits imprimés.

Configurabilité et personnalisation

Les modules ESP32 peuvent être commandés avec différentes configurations d'antenne (par exemple, antenne PCB, connecteur d'antenne) et tailles de flash, afin de répondre aux besoins de différentes applications. Les modules ESP32 offrent également des personnalisations de fabrication avec des microprogrammes d'application préprogrammés, des données personnalisées et des certificats en nuage prévisionnés.

Prêt pour l'application

Tous les modules de la série ESP32 ont une large plage de température de fonctionnement de -40°C à 105°C, et sont adaptés au développement d'applications commerciales grâce à une conception robuste à 4 couches entièrement conforme aux normes FCC, CE-RED, SRRC, IC, KCC et TELEC.

partie 2

Programmer la brique magnetics

fiches d'activité



MAGNETICS

Créer des projets multicartes



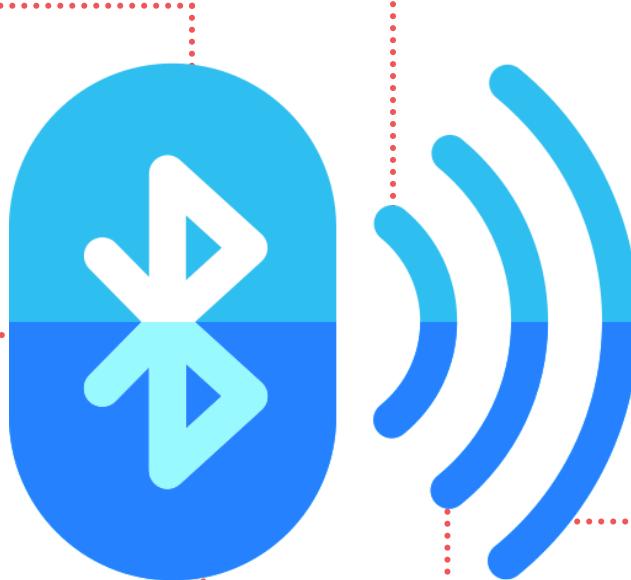
Disponible sur

Durée

50 minutes

Matériel

- 4 cartes programmables "STM32 IoT Node Board"
- 1 câble USB Micro-B
- 1 écran OLED Monochrome 1.3" 128x64 OLED de Adafruit
- 1 câble QT pour connecter l'écran à la carte



De quoi parle-t-on ?

Dans cette activité, nous allons programmer plusieurs cartes électroniques et échanger des données de capteurs à l'aide de l'extension Magnetics, qui permet une communication maillée sans fil.

Coopération



Niveau de difficulté

Avancé

OBJECTIFS D'APPRENTISSAGE

- Utiliser l'extension magnetics pour constituer des projets multicartes
- Appréhender le fonctionnement des modules bluetooth BLE Mesh

MAGNETICS - CRÉER DES PROJETS MULTICARTES



Cette fiche d'activité propose de créer des projets plus complexes en utilisant plusieurs cartes électroniques non connectées entre elles. Une fois les capteurs maîtrisés, nous pouvons en effet mettre en place des expériences nécessitant l'utilisation de plusieurs cartes. Afin de réaliser la collecte des données, il faut pouvoir faire communiquer les cartes entre elles par les airs. Dans cette activité nous allons programmer plusieurs cartes électroniques et échanger des données de capteurs à l'aide de l'extension **Magnetics** que permet de mettre en œuvre une communication sans fil maillée. Le projet magnetics prend la forme d'une brique technique logicielle implantée directement dans MakeCode. Ce développement est basé sur l'utilisation de la technologie de réseau maillé **Bluetooth Low Energy Mesh** (BLE Mesh) compatible avec toutes les cartes programmables disposant d'un module **Bluetooth Low Energy**.

Ressources : <https://www.magnetics.edu-up.fr/>

<https://blog.rtone.fr/bluetooth-mesh>

https://fr.wikipedia.org/wiki/Bluetooth_%C3%A0_basse_consommation



ÉTAPE 1 - CONSTRUIRE



Pour réaliser cette activité nous avons besoin de **quatre cartes STM32 IoT Nodes**. Trois d'entre elles seront **émettrices** de données de capteurs (température, humidité, pression), et la dernière sera **collectrice** des données qu'elle affichera sur un écran OLED. Mis à part l'écran de la dernière, il n'y a pas de câblage car nous utiliserons uniquement les capteurs internes. Nous allons donc vous donner la marche à suivre pour câbler et programmer en premier lieu la carte collectrice puis dans un second temps, programmer individuellement chaque carte émettrice afin de pouvoir construire votre projet.

Activité 1 - Préparer, câbler et programmer la carte collectrice

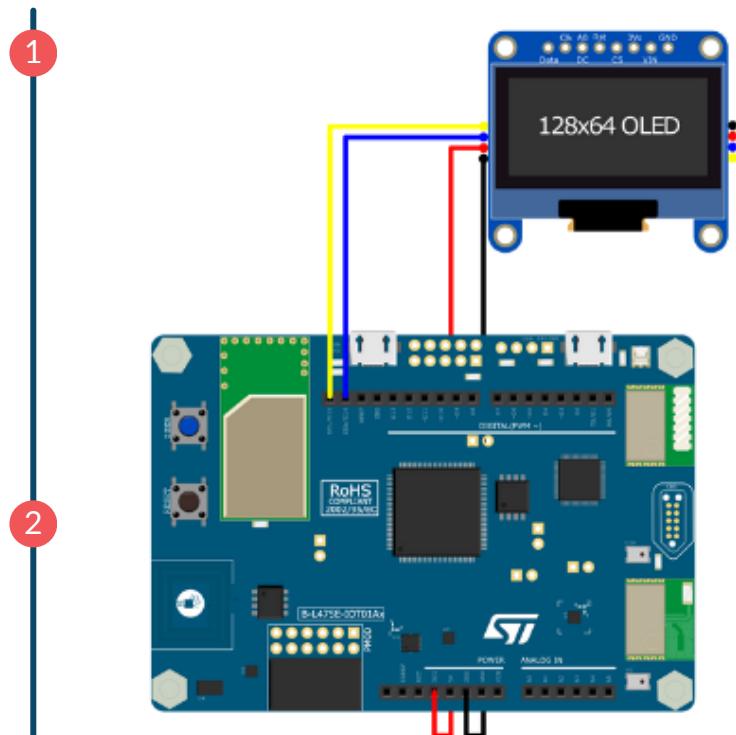
Câbler l'écran OLED

Nous devons en premier lieu câbler l'écran OLED directement à la carte collectrice. Il y a deux façons de câbler l'écran **OLED SSD1306** à une carte, soit avec une connexion **I2C** ou **SPI**. Pour notre écran, nous utilisons la connexion **I2C** via le câble **QWIIC/STEMMA** avec la convention suivante :

- **Noir** pour **GND**
- **Rouge** pour **V+ (3V3)**
- **Bleu** pour **SDA (D14)**
- **Jaune** pour **SCL (D15)**

Connecter la carte à l'ordinateur

Avec votre câble USB, connectez la carte collectrice à votre ordinateur en utilisant le connecteur micro-USB. Si tout se passe bien, vous devriez voir apparaître sur votre ordinateur un nouveau lecteur.



Câblage de l'écran OLED sur la carte collectrice

MAGNETICS - CRÉER DES PROJETS MULTICARTES



ÉTAPE 1 - CONSTRUIRE



Ouvrir MakeCode

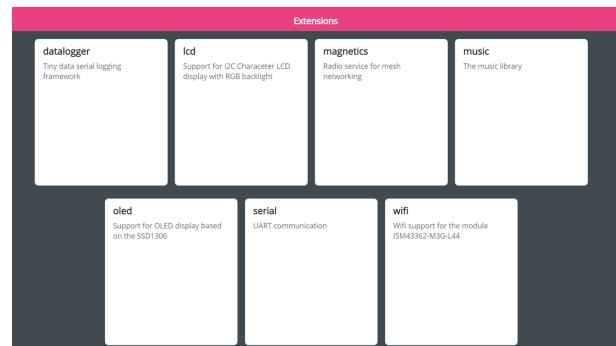
Allez dans [l'éditeur MakeCode de Let's STEAM](#). Sur la page d'accueil, créez un nouveau projet en cliquant sur le bouton "Nouveau projet". Donnez à votre projet un nom plus expressif que "Sans titre" et lancez votre éditeur.

Ressource : makecode.lets-steam.eu

Installer les extensions

Après avoir créé votre nouveau projet, vous obtiendrez l'écran par défaut et vous devrez installer deux extensions pour ce projet spécifique : "magnetics" et "oled".

3



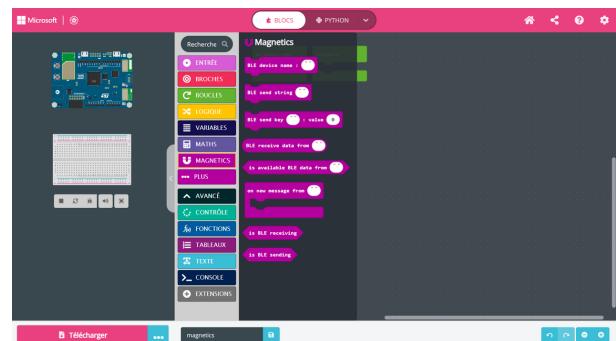
Liste d'extensions et outil de recherche

i Les extensions dans MakeCode sont des groupes de blocs de code qui ne sont pas directement inclus dans les blocs de code de base que l'on trouve dans MakeCode. Les extensions, comme leur nom l'indique, ajoutent des blocs pour des fonctionnalités spécifiques. Il existe des extensions pour un large éventail de fonctionnalités très utiles, ajoutant des capacités de manette de jeu, de clavier, de souris, de servomoteurs, de la robotique et bien plus encore.

Vous voyez le bouton noir **AVANCÉ** en bas de la colonne des différents groupes de blocs. En cliquant sur ce bouton, vous verrez apparaître des groupes de blocs supplémentaires. En bas, il y a une boîte grise appelée **EXTENSIONS**. Cliquez sur ce bouton.

Dans la liste des extensions disponibles, vous pouvez trouver l'extension **Magnetics** qui sera utilisée pour cette activité. Si elle n'est pas directement disponible sur votre écran, vous pouvez la rechercher en utilisant la barre de recherche. Cliquez sur l'extension et un nouveau groupe de blocs apparaîtra sur l'écran principal. Répétez cette action pour installer l'extension **OLEO** également disponible. Vous pouvez commencer à programmer.

4

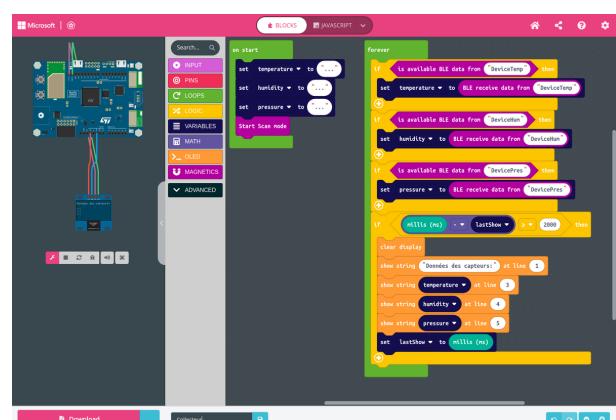


Blocs associés à l'extension magnetics

Programmer la carte

Dans l'éditeur JavaScript de MakeCode, copiez/collez le code disponible dans la section "**Programmer**" ci-dessous. Cliquez sur le bouton "**Télécharger**" et attendez que la carte finisse de clignoter.

5



Capture d'écran de MakeCode avec les blocs du programme du collecteur



ETAPE 1 - CONSTRUIRE



Activité 2 - Programmer chaque carte émettrice individuellement

Une fois la carte collectrice câblée avec l'écran OLED et programmée, nous pouvons préparer les trois cartes émettrices en suivant les mêmes étapes de programmation que lors de l'étape 1. Pour chaque carte, il faudra donc effectuer les tâches suivantes :

Connecter la carte à l'ordinateur

Avec votre câble USB, connectez la carte émettrice que vous souhaitez programmer à votre ordinateur en utilisant le **connecteur micro-USB**. Si tout se passe bien, vous devriez voir apparaître sur votre ordinateur un nouveau lecteur. Ce lecteur est utilisé pour programmer la carte en copiant simplement un fichier binaire.

6

Ouvrir MakeCode

Allez dans [l'éditeur MakeCode de Let's STEAM](#). Sur la page d'accueil, créez un nouveau projet en cliquant sur le bouton "**Nouveau projet**". Donnez à votre projet un nom plus expressif que "Sans titre" et lancez votre éditeur.

Ressource : makecode.lets-steam.eu

7

Installer l'extension

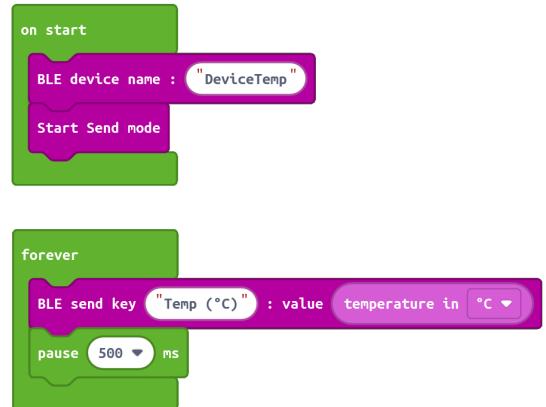
De la même manière qu'à l'étape 1, ajoutez l'extension magnetics à la liste de bloc via le menu "**AVANCÉ**" puis "**EXTENSIONS**".

8

Programmer la carte

Dans l'éditeur JavaScript de MakeCode, copiez/collez le code disponible dans la section "**Programmer**" ci-dessous relatif à la carte émettrice que vous êtes en train de programmer. Si ce n'est pas déjà fait, pensez à donner un nom à votre projet et cliquez sur le bouton "**Télécharger**" et attendez que la carte finisse de clignoter.

9



Exemple de la programmation par bloc relative à la carte émettrice collectant les données de température

Activité 3 - Exécuter, modifier, jouer

Une fois votre carte collectrice reliée à votre écran, et vos quatre cartes programmées, votre programme est prêt à être utilisé. Votre programme s'exécutera automatiquement chaque fois que vous les mettrez sous tension ou que vous réinitialiserez votre carte (appuyez sur le bouton intitulé RESET). Essayez de comprendre le fonctionnement de votre code et commencez à le modifier en créant vos propres projets.

MAGNETICS - CRÉER DES PROJETS MULTICARTES



ETAPE 2 - PROGRAMMER



Code de la carte collectrice

```

let temperature = "...";
let humidity = "...";
let pressure = "...";
let lastShow = 0;

magnetics.startScanning()
forever(function () {

    if (magnetics.availableDataFromName("DeviceTemp")) {
        temperature = magnetics.readDataFromName("DeviceTemp")
    }
    if (magnetics.availableDataFromName("DeviceHum")) {
        humidity = magnetics.readDataFromName("DeviceHum")
    }
    if (magnetics.availableDataFromName("DevicePres")) {
        pressure = magnetics.readDataFromName("DevicePres")
    }

    if( control.millis() - lastShow >= 2000 ){
        oled.clear()
        oled.showString("Données des capteurs:", 1)
        oled.showString(temperature, 3)
        oled.showString(humidity, 4)
        oled.showString(pressure, 5)

        lastShow = control.millis();
    }
})

```

Code de la carte émettrice de la température

```

magnetics.setLocalName("DeviceTemp")
magnetics.startEmitting()
forever(function () {
    magnetics.setAdvertisingKeyValueData("Temp ( °C)", input.temperature(TemperatureUnit.Celsius))
    pause(500)
})

```



ETAPE 2 - PROGRAMMER



Code de la carte émettrice de l'humidité

```
magneticss.setLocalName( "DeviceHum" )
magneticss.startEmitting()
forever(function () {
  magneticss.setAdvertisingKeyValueData( "Hum (%)", input.humidity())
  pause(500)
})
```

Code de la carte émettrice de la pression

```
magneticss.setLocalName( "DevicePres" )
magneticss.startEmitting()
forever(function () {
  magneticss.setAdvertisingKeyValueData( "Pres (hPa)", input.pressure(PressureUnit.HectoPascal))
  pause(500)
})
```

Comment cela fonctionne-t-il ? Initialiser la collecte de données :

Cela fait beaucoup de code, mais rien de très complexe, puisque le code des **cartes émettrices** est quasiment identique (il n'y a que le capteur qui change) et reste simple à comprendre.

Pour commencer, on donne un nom à notre carte à l'aide de la fonction `setLocalName`, ce qui permettra au collecteur de reconnaître la donnée envoyée. Pour que la carte puisse émettre des données, il faut lui préciser son rôle, c'est ce que fait la fonction `startEmitting`. Enfin, via la fonction `setAdvertisingKeyValueData`, nous envoyons la donnée du capteur toutes les 500 millisecondes (c'est-à-dire deux fois par seconde).

Enfin, intéressons-nous au code du collecteur. Au début, nous déclarons 4 variables:

- `temperature`, `humidity` et `pressure` qui contiendront les données émises par les émetteurs (respectivement pour la température, l'humidité et la pression atmosphérique)
- `lastShow` qui permet de savoir à quand remonte le dernier affichage des données (en millisecondes).

Une fois les variables initialisées, nous faisons appel à `startScanning`, pour rechercher les cartes émettrices qui sont à proximité. Le reste du code se trouve dans la fonction `forever` ce qui implique, que nous allons répéter indéfiniment le code suivant. Les trois premiers `if(...)` permettent de savoir si nous avons reçu des données de la part des émetteurs (grâce à la fonction `availableDataFromName`), si tel est le cas, alors on enregistre cette donnée dans la variable associée (en utilisant `readDataFromName`).

Maintenant que nous avons les données, il faut les afficher sur notre écran OLED, c'est le rôle de la dernière condition `if`, qui nous permet de mettre à jour notre écran avec les nouvelles données, seulement si cela fait plus de deux seconds qu'il n'a pas été rafraîchi.

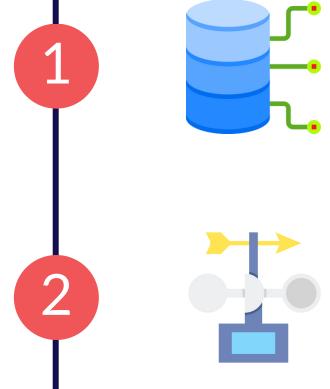
MAGNETICS - CRÉER DES PROJETS MULTICARTES



ETAPE 3 - AMÉLIORER

Cette activité ne fait qu'afficher les informations sur un écran, mais il est possible d'utiliser le datalogger, pour enregistrer nos données et les analyser.

Utiliser une boîte étanche et faire une station météo, la connexion étant sans fil, on pourrait avoir un écran en intérieur nous affichant les données environnementales de l'extérieur.



ALLER PLUS LOIN

Bluetooth - Découvrir plus d'informations sur cette norme de télécommunication si présente dans notre vie quotidienne

<https://fr.wikipedia.org/wiki/Bluetooth>



CircuitPython BLE Libraries on Any Computer -

Utilisez le code BLE de CircuitPython sur les ordinateurs de bureau, les ordinateurs portables et les Raspberry Pi grâce aux bibliothèques Adafruit

<https://learn.adafruit.com/circuitpython-bluetooth-libraries-on-any-computer>



Bouton de volume BLE avec CircuitPython -

Transformez votre Bluefruit de Circuit Playground en bouton de volume BLE sans fil

<https://learn.adafruit.com/bluetooth-le-hid-volume-knob-with-circuitpython>



Explorer d'autres fiches d'activité du projet Let's STEAM

R1AS15 - Collecter les données

https://github.com/letssteam/Resources/blob/main/3_AS_Programming/FR/LS_R1AS15_Data_FR.pdf



R1AS11 - Fabriquer un thermomètre

https://github.com/letssteam/Resources/blob/main/3_AS_Programming/FR/LS_R1AS11_Termometer_FR.pdf



MAGNETICS

Créer des projets multicartes



Disponible sur

Durée

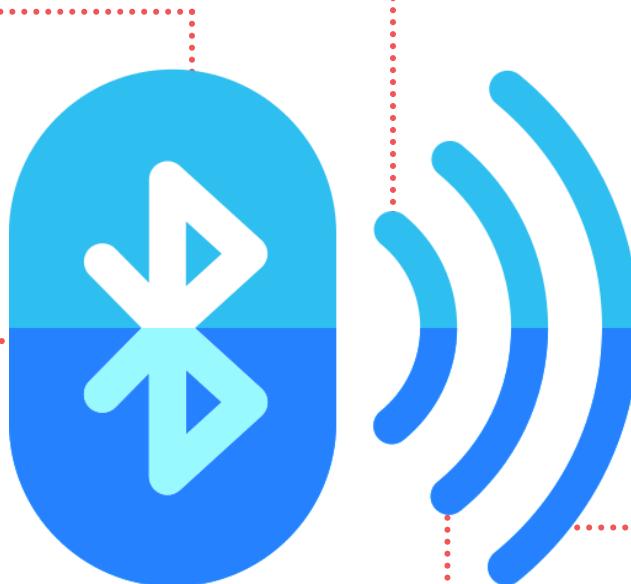
50 minutes

Matériel

- 4 cartes programmables "STM32 IoT Node Board"
- 1 câble USB Micro-B
- 1 écran OLED Monochrome 1.3" 128x64 OLED de Adafruit
- 1 câble QT pour connecter l'écran à la carte

De quoi parle-t-on ?

Dans cette activité, nous allons programmer plusieurs cartes électroniques et échanger des données de capteurs à l'aide de l'extension Magnetics, qui permet une communication maillée sans fil.



Coopération



Niveau de difficulté

Avancé

OBJECTIFS D'APPRENTISSAGE

- Utiliser l'extension magnetics pour constituer des projets multicartes
- Appréhender le fonctionnement des modules bluetooth BLE Mesh



Cette fiche d'activité propose de créer des projets plus complexes en utilisant plusieurs cartes électroniques non connectées entre elles. Une fois les capteurs maîtrisés, nous pouvons en effet mettre en place des expériences nécessitant l'utilisation de plusieurs cartes. Afin de réaliser la collecte des données, il faut pouvoir faire communiquer les cartes entre elles par les airs. Dans cette activité nous allons programmer plusieurs cartes électroniques et échanger des données de capteurs à l'aide de l'extension **Magnetics** que permet de mettre en œuvre une communication sans fil maillée. Le projet magnetics prend la forme d'une brique technique logicielle implantée directement dans MakeCode. Ce développement est basé sur l'utilisation de la technologie de réseau maillé **Bluetooth Low Energy Mesh** (BLE Mesh) compatible avec toutes les cartes programmables disposant d'un module **Bluetooth Low Energy**.

Ressources : <https://www.magnetics.edu-up.fr/>

<https://blog.rtone.fr/bluetooth-mesh>

https://fr.wikipedia.org/wiki/Bluetooth_%C3%A0_basse_consommation



ÉTAPE 1 - CONSTRUIRE



Pour réaliser cette activité nous avons besoin de **quatre cartes STM32 IoT Nodes**. Trois d'entre elles seront **émettrices** de données de capteurs (température, humidité, pression), et la dernière sera **collectrice** des données qu'elle affichera sur un écran OLED. Mis à part l'écran de la dernière, il n'y a pas de câblage car nous utiliserons uniquement les capteurs internes. Nous allons donc vous donner la marche à suivre pour câbler et programmer en premier lieu la carte collectrice puis dans un second temps, programmer individuellement chaque carte émettrice afin de pouvoir construire votre projet.

Activité 1 - Préparer, câbler et programmer la carte collectrice

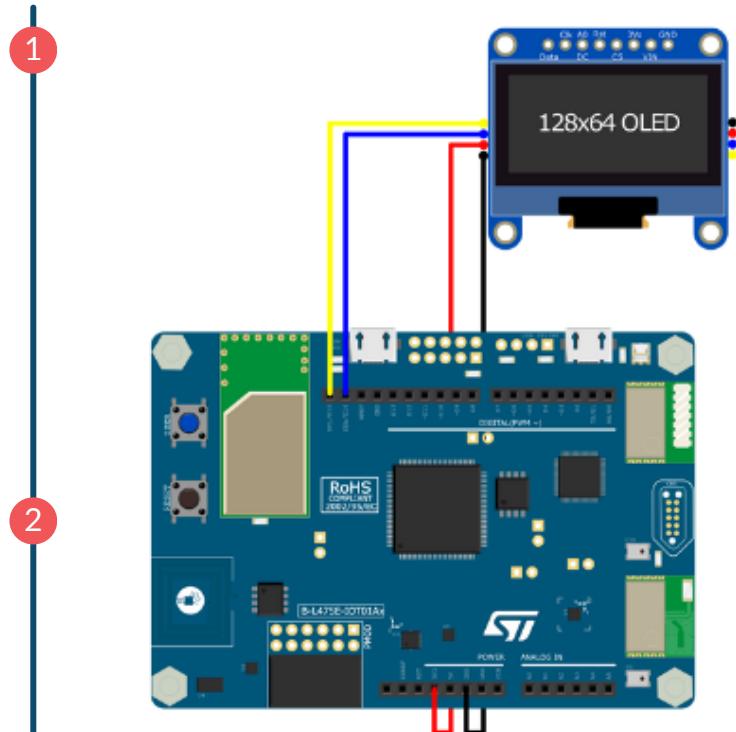
Câbler l'écran OLED

Nous devons en premier lieu câbler l'écran OLED directement à la carte collectrice. Il y a deux façons de câbler l'écran **OLED SSD1306** à une carte, soit avec une connexion **I2C** ou **SPI**. Pour notre écran, nous utilisons la connexion **I2C** via le câble **QWIIC/STEMMA** avec la convention suivante :

- **Noir** pour **GND**
- **Rouge** pour **V+ (3V3)**
- **Bleu** pour **SDA (D14)**
- **Jaune** pour **SCL (D15)**

Connecter la carte à l'ordinateur

Avec votre câble USB, connectez la carte collectrice à votre ordinateur en utilisant le connecteur micro-USB. Si tout se passe bien, vous devriez voir apparaître sur votre ordinateur un nouveau lecteur.



Câblage de l'écran OLED sur la carte collectrice

MAGNETICS - CRÉER DES PROJETS MULTICARTES



ÉTAPE 1 - CONSTRUIRE

Ouvrir l'éditeur Micropython

Allez dans l'éditeur Micropython (<https://python.lets-steam.eu/>). Depuis cet éditeur, vous pourrez directement copier/coller votre code.

Programmer la carte

Dans l'éditeur, copiez/collez le code disponible dans la section "**Programmer**" ci-dessous. Cliquez sur le bouton "**flash or download**" et attendez que la carte finisse de clignoter.



```
1 from machine import Pin, I2C
2
3 import ssd1306
4 import magne
5 import time
6
7 oled = ssd1306.SSD1306_I2C(I2C(1))
8 mag = magne.Magnete(I2C(1))
9
10 lastShow = 0
11 temperature = ".,.," 
12 humidity = ".,.," 
13 pressure = ".,.," 
14
15 mag.start_scanning()
16
17 while True: # Creation d'une boucle "infinie" (pas de cléuse de sortie)
18     if mag.available_data_from_name("DeviceTemp"):
19         temperature = mag.read_data_from_name("DeviceTemp")
20
21     if mag.available_data_from_name("DeviceHumid"):
22         humidity = mag.read_data_from_name("DeviceHumid")
23
24     if mag.available_data_from_name("DevicePres"):
25         pressure = mag.read_data_from_name("DevicePres")
26
27     time.sleep(1) # laisser temps pour lire les données
28
29     oled.text("Données des capteurs:", 0, 0)
30     oled.text("Température: " + str(temperature), 0, 10)
31     oled.text("Humidité: " + str(humidity), 0, 20)
32     oled.text("pression: " + str(pressure), 0, 30)
33
34     lastShow = time.time()
35
36     lastShow = time.time_ns()
```

Activité 2 - Programmer chaque carte émettrice individuellement

Une fois la carte collectrice câblée avec l'écran OLED et programmée, nous pouvons préparer les trois cartes émettrices en suivant les mêmes étapes de programmation que lors de l'étape 1. Pour chaque carte, il faudra donc effectuer les tâches suivantes :

Connecter la carte à l'ordinateur

Avec votre câble USB, connectez la carte émettrice que vous souhaitez programmer à votre ordinateur en utilisant le connecteur **micro-USB**. Si tout se passe bien, vous devriez voir apparaître sur votre ordinateur un nouveau lecteur. Ce lecteur est utilisé pour programmer la carte en copiant simplement un fichier binaire.

Ouvrir l'éditeur Micropython

Allez dans l'éditeur MicroPython (<https://python.lets-steam.eu/>). Depuis cet éditeur, vous pourrez directement copier/coller votre code.

Programmer la carte

Dans l'éditeur, copiez/collez le code disponible dans la section "**Programmer**" ci-dessous. Cliquez sur le bouton "**flash or download**" et attendez que la carte finisse de clignoter.

Activité 3 - Exécuter, modifier, jouer

Une fois votre carte collectrice reliée à votre écran, et vos quatre cartes programmées, votre programme est prêt à être utilisé. Votre programme s'exécutera automatiquement chaque fois que vous les mettrez sous tension ou que vous réinitialiserez votre carte (appuyez sur le bouton intitulé RESET). Essayez de comprendre le fonctionnement de votre code et commencez à le modifier en créant vos propres projets.

ETAPE 2 - PROGRAMMER



Code de la carte collectrice

```
from machine import Pin, I2C
import ssd1306
import magnetics
import time

oled = ssd1306.SSD1306_I2C(I2C(1))
mag = magnetics.Magnetics(I2C(2))

lastShow = 0
temperature = "..."
humidity = "..."
pressure = "..."
mag.start_scanning()

while True: # Création d'une boucle "infinie" (pas de clause de sortie)
    if mag.available_data_from_name("DeviceTemp"):
        temperature = mag.read_data_from_name("DeviceTemp")

    if mag.available_data_from_name("DeviceHum"):
        humidity = mag.read_data_from_name("DeviceHum")

    if mag.available_data_from_name("DevicePres"):
        pressure = mag.read_data_from_name("DevicePres")

    if time.ticks_ms() - lastShow >= 2000:
        oled.clear()
        oled.text("Données des capteurs:", 0, 0)
        oled.text(temperature, 0, 16)
        oled.text(humidity, 0, 24)
        oled.text(pressure, 0, 32)
        oled.show()
        lastShow = time.ticks_ms()
```

ETAPE 2 - PROGRAMMER



Code de la carte émettrice de la température

```
from machine import Pin, I2C
import magnetics
import time

mag = magnetics.Magnetics(I2C(2))
mag.setLocalName("DeviceTemp")
mag.startEmitting()

while True: # Création d'une boucle "infinie" (pas de clause de sortie)
    magnetics.setAdvertisingKeyValueData("Temp (°C)", input.temperature(TemperatureUnit.Celsius))
    pause(500)
```

Code de la carte émettrice de l'humidité

```
from machine import Pin, I2C
import magnetics
import time

mag = magnetics.Magnetics(I2C(2))
mag.setLocalName("DeviceHum")
mag.startEmitting()

while True: # Création d'une boucle "infinie" (pas de clause de sortie)
    magnetics.setAdvertisingKeyValueData("Hum (%)", input.temperature(TemperatureUnit.Celsius))
    pause(500)
```

Code de la carte émettrice de la pression

```
from machine import Pin, I2C
import magnetics
import time

mag = magnetics.Magnetics(I2C(2))
mag.setLocalName("DevicePres")
mag.startEmitting()

while True: # Création d'une boucle "infinie" (pas de clause de sortie)
    magnetics.setAdvertisingKeyValueData("Pres (hPa)", input.temperature(TemperatureUnit.Celsius))
    pause(500)
```

ETAPE 2 - PROGRAMMER



Comment cela fonctionne-t-il ? Initialiser la collecte de données :

Cela fait beaucoup de code, mais rien de très complexe, puisque le code des **cartes émettrices** est quasiment identique (il n'y a que le capteur qui change) et reste simple à comprendre.

Pour commencer, on donne un nom à notre carte à l'aide de la fonction `setLocalName`, ce qui permettra au collecteur de reconnaître la donnée envoyée. Pour que la carte puisse émettre des données, il faut lui préciser son rôle, c'est ce que fait la fonction `startEmitting`. Enfin, via la fonction `setAdvertisingKeyValueData`, nous envoyons la donnée du capteur toutes les 500 millisecondes (c'est-à-dire deux fois par seconde).

Enfin, intéressons-nous au code du collecteur. Au début, nous déclarons 4 variables:

- `temperature`, `humidity` et `pressure` qui contiendrons les données émises par les émetteurs (respectivement pour la température, l'humidité et la pression atmosphérique)
- `lastShow` qui permet de savoir à quand remonte le dernier affichage des données (en millisecondes).

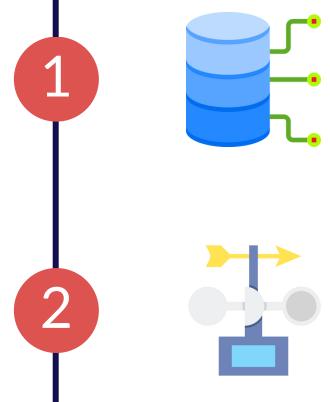
Une fois les variables initialisées, nous faisons appel à `startScanning`, pour rechercher les cartes émettrices qui sont à proximité.

Le reste du code se trouve dans la boucle principale ce qui implique, que nous allons répéter indéfiniment le code suivant. Les trois premiers `if(...)` permettent de savoir si nous avons reçu des données de la part des émetteurs (grâce à la fonction `availableDataFromName`), si tel est le cas, alors on enregistre cette donnée dans la variable associée (en utilisant `readDataFromName`).

Maintenant que nous avons les données, il faut les afficher sur notre écran OLED, c'est le rôle de la dernière condition `if`, qui nous permet de mettre à jour notre écran avec les nouvelles données, seulement si cela fait plus de deux seconds qu'il n'a pas été rafraîchi.

ETAPE 3 - AMÉLIORER

Cette activité ne fait qu'afficher les informations sur un écran, mais il est possible d'utiliser le **datalogger**, pour enregistrer nos données et les analyser.



Utiliser une boîte étanche et faire une **station météo**, la connexion étant sans fil, on pourrait avoir un écran en intérieur nous affichant les **données environnementales de l'extérieur**.

ALLER PLUS LOIN

Bluetooth - Découvrir plus d'informations sur cette norme de télécommunication si présente dans notre vie quotidienne

<https://fr.wikipedia.org/wiki/Bluetooth>



CircuitPython BLE Libraries on Any Computer - Utilisez le code BLE de CircuitPython sur les ordinateurs de bureau, les ordinateurs portables et les Raspberry Pi grâce aux bibliothèques Adafruit

<https://learn.adafruit.com/circuitpython-ble-libraries-on-any-computer>



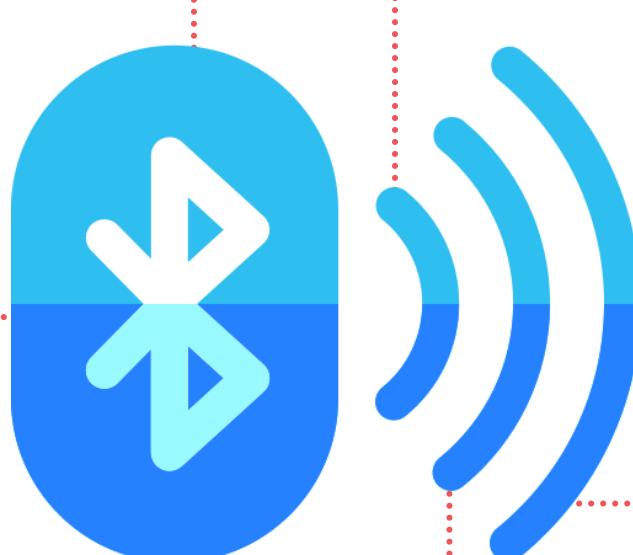
Bouton de volume BLE avec CircuitPython - Transformez votre Bluefruit de Circuit Playground en bouton de volume BLE sans fil

<https://learn.adafruit.com/bluetooth-le-hid-volume-knob-with-circuitpython>



MAGNETICS

Créer des projets multicartes

**Disponible sur****Durée**

50 minutes

Matériel

- 4 cartes programmables "Micro:bit v2"
- 1 câble USB Micro-B
- 1 écran OLED Monochrome 1.3" 128x64 OLED de Adafruit
- 1 câble QT pour connecter l'écran à la carte

De quoi parle-t-on ?

Dans cette activité, nous allons programmer plusieurs cartes électroniques et échanger des données de capteurs à l'aide de l'extension Magnetics, qui permet une communication maillée sans fil.

Coopération**Niveau de difficulté**

Avancé

OBJECTIFS D'APPRENTISSAGE

- Utiliser l'extension magnetics pour constituer des projets multicartes
- Appréhender le fonctionnement des modules bluetooth BLE Mesh



MAGNETICS - CRÉER DES PROJETS MULTICARTES



Cette fiche d'activité propose de créer des projets plus complexes en utilisant plusieurs cartes électroniques non connectées entre elles. Une fois les capteurs maîtrisés, nous pouvons en effet mettre en place des expériences nécessitant l'utilisation de plusieurs cartes. Afin de réaliser la collecte des données, il faut pouvoir faire communiquer les cartes entre elles par les airs. Dans cette activité nous allons programmer plusieurs cartes électroniques et échanger des données de capteurs à l'aide de l'extension **Magnetics** que permet de mettre en œuvre une communication sans fil maillée. Le projet magnetics prend la forme d'une brique technique logicielle implantée directement dans MakeCode. Ce développement est basé sur l'utilisation de la technologie de réseau maillé **Bluetooth Low Energy Mesh** (BLE Mesh) compatible avec toutes les cartes programmables disposant d'un module **Bluetooth Low Energy**.

Ressources : <https://www.magnetics.edu-up.fr/>
<https://blog.rtone.fr/bluetooth-mesh>
https://fr.wikipedia.org/wiki/Bluetooth_%C3%A0_basse_consommation



ÉTAPE 1 - CONSTRUIRE



Pour réaliser cette activité nous avons besoin de **quatre cartes micro:bit v2**. Trois d'entre elles seront **émettrices** de données de capteurs (température, humidité, pression), et la dernière sera **collectrice** des données qu'elle affichera sur ses LEDs. Nous utiliserons uniquement les capteurs internes. Nous allons donc vous donner la marche à suivre pour câbler et programmer en premier lieu la carte collectrice puis dans un second temps, programmer individuellement chaque carte émettrice afin de pouvoir construire votre projet.

Activité 1 - Préparer, câbler et programmer la carte collectrice

Connecter la carte à l'ordinateur

Avec votre câble USB, connectez la carte collectrice à votre ordinateur en utilisant le connecteur micro-USB. Si tout se passe bien, vous devriez voir apparaître sur votre ordinateur un nouveau lecteur.

Ouvrir MakeCode

Allez dans **l'éditeur MakeCode pour micro:bit**. Sur la page d'accueil, créez un nouveau projet en cliquant sur le bouton **"Nouveau projet"**. Donnez à votre projet un nom plus expressif que "Sans titre" et lancez votre éditeur.

Ressource : <https://makecode.microbit.org/>

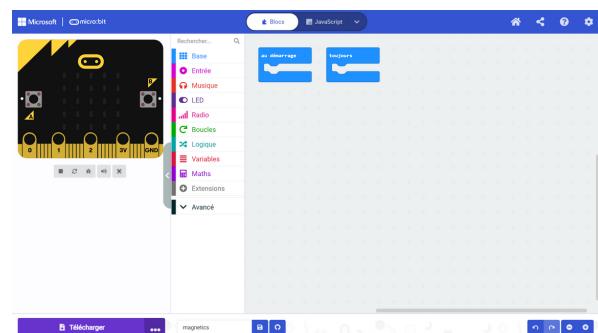
Installer l'extension

Après avoir créé votre nouveau projet, vous obtiendrez l'écran par défaut "prêt à l'emploi" et vous devrez installer une extension.

1

2

3



Éditeur MakeCode pour micro:bit

MAGNETICS - CRÉER DES PROJETS MULTICARTES



ÉTAPE 1 - CONSTRUIRE



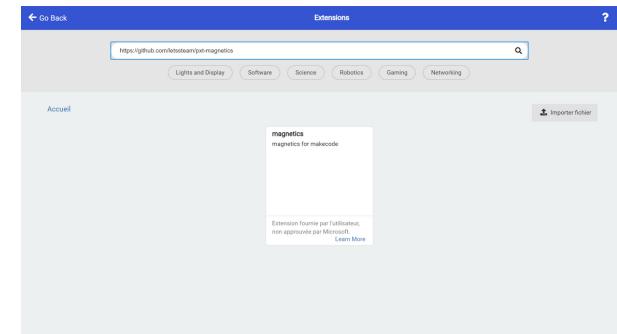
Les extensions dans MakeCode sont des groupes de blocs de code qui ne sont pas directement inclus dans les blocs de code de base que l'on trouve dans MakeCode. Les extensions, comme leur nom l'indique, ajoutent des blocs pour des fonctionnalités spécifiques. Il existe des extensions pour un large éventail de fonctionnalités très utiles, ajoutant des capacités de manette de jeu, de clavier, de souris, de servomoteurs, de la robotique et bien plus encore.

Cliquez sur le bloc "EXTENSIONS" pour installer les nouveaux blocs dont vous allez avoir besoin. Pour accéder à magnetics, vous devrez copier/coller l'URL suivante dans la barre de recherche : <https://github.com/letssteam/pxt-magnetics>.

Magnetics utilisant le bluetooth, MakeCode vous demandera de supprimer l'extension "RADIO" de votre projet.

Programmer la carte

Dans l'éditeur JavaScript de MakeCode, copiez/collez le code disponible dans la section "Programmer" ci-dessous. Cliquez sur le bouton "Télécharger" et attendez que la carte finisse de clignoter.



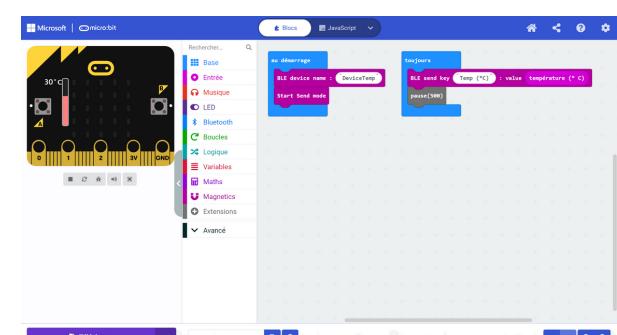
Extension magnetics

Activité 2 - Programmer chaque carte émettrice individuellement

Une fois la carte collectrice programmée, nous pouvons préparer les trois cartes émettrices en suivant les mêmes étapes de programmation que lors de l'étape 1. Pour chaque carte, il faudra donc effectuer les tâches suivantes :

Connecter la carte à l'ordinateur

Avec votre câble USB, connectez la carte émettrice que vous souhaitez programmer à votre ordinateur en utilisant le **connecteur micro-USB**. Si tout se passe bien, vous devriez voir apparaître sur votre ordinateur un nouveau lecteur. Ce lecteur est utilisé pour programmer la carte en copiant simplement un fichier binaire.



Exemple de l'éditeur en mode bloc pour le code de la carte émettrice de la température

MAGNETICS - CRÉER DES PROJETS MULTICARTES



ETAPE 1 - CONSTRUIRE

Ouvrir MakeCode

Allez dans [l'éditeur MakeCode pour micro:bit](#). Sur la page d'accueil, créez un nouveau projet en cliquant sur le bouton "Nouveau projet". Donnez à votre projet un nom plus expressif que "Sans titre" et lancez votre éditeur.

Ressource : <https://makecode.microbit.org/>

Installer l'extension

De la même manière qu'à l'étape 1, ajoutez l'extension **magnetics** à la liste de bloc via le menu "**EXTENSIONS**".

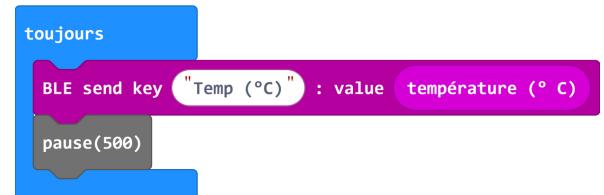
Programmer la carte

Dans l'éditeur JavaScript de MakeCode, copiez/collez le code disponible dans la section "**Programmer**" ci-dessous relatif à la carte émettrice que vous êtes en train de programmer. Si ce n'est pas déjà fait, pensez à donner un nom à votre projet et cliquez sur le bouton "**Télécharger**" et attendez que la carte finisse de clignoter.

6



7



8

Blocs pour le code de la carte émettrice de la température

Activité 3 - Exécuter, modifier, jouer

Une fois votre carte collectrice reliée à votre écran, et vos quatre cartes programmées, votre programme est prêt à être utilisé. Votre programme s'exécutera automatiquement chaque fois que vous les mettrez sous tension ou que vous réinitialiserez votre carte (appuyez sur le bouton intitulé RESET). Essayez de comprendre le fonctionnement de votre code et commencez à le modifier en créant vos propres projets.

MAGNETICS - CRÉER DES PROJETS MULTICARTES

ETAPE 2 - PROGRAMMER



Code de la carte collectrice

```

let temperature = "...";
let lightLevel = "...";
let sound = "...";
let lastShow = 0;

magnetics.startScanning()
forever(function () {

    if (magnetics.availableDataFromName("DeviceTemp")) {
        temperature = magnetics.readDataFromName("DeviceTemp")
    }
    if (magnetics.availableDataFromName("DeviceLight")) {
        lightLevel = magnetics.readDataFromName("DeviceLight")
    }
    if (magnetics.availableDataFromName("DeviceSound")) {
        sound = magnetics.readDataFromName("DeviceSound")
    }

    if( control.millis() - lastShow >= 2000 ){
        basic.showString("Capteurs:")
        basic.showString(temperature)
        basic.showString(lightLevel)
        basic.showString(sound)

        lastShow = control.millis();
    }
})

```

Code de la carte émettrice de la température

```

magnetics.setLocalName("DeviceTemp")
magnetics.startEmitting()
basic.forever(function () {
    magnetics.setAdvertisingKeyValueData("Temp ( °C)", input.temperature())
    pause(500)
})

```

MAGNETICS - CRÉER DES PROJETS MULTICARTES



ETAPE 2 - PROGRAMMER



Code de la carte émettrice de l'intensité lumineuse

```
magnetics.setLocalName("DeviceLight")
magnetics.startEmitting()
forever(function () {
  magnetics.setAdvertisingKeyValueData("Light()", input.lightLevel())
  pause(500)
})
```

Code de la carte émettrice du niveau sonore

```
magnetics.setLocalName("DeviceSound")
magnetics.startEmitting()
forever(function () {
  magnetics.setAdvertisingKeyValueData("Sound()", input.soundLevel())
  pause(500)
})
```

Comment cela fonctionne-t-il ? Initialiser la collecte de données :

Cela fait beaucoup de code, mais rien de très complexe, puisque le code des **cartes émettrices** est quasiment identique (il n'y a que le capteur qui change) et reste simple à comprendre.

Pour commencer, on donne un nom à notre carte à l'aide de la fonction `setLocalName`, ce qui permettra au collecteur de reconnaître la donnée envoyée. Pour que la carte puisse émettre des données, il faut lui préciser son rôle, c'est ce que fait la fonction `startEmitting`. Enfin, via la fonction `setAdvertisingKeyValueData`, nous envoyons la donnée du capteur toutes les 500 millisecondes (c'est-à-dire deux fois par seconde).

Enfin, intéressons-nous au code du collecteur. Au début, nous déclarons 4 variables:

- `temperature`, `light` et `sound` qui contiendrons les données émises par les émetteurs (respectivement pour la température, l'intensité lumineuse et le niveau sonore)
- `lastShow` qui permet de savoir à quand remonte le dernier affichage des données (en millisecondes).

Une fois les variables initialisées, nous faisons appel à `startScanning`, pour rechercher les cartes émettrices qui sont à proximité. Le reste du code se trouve dans la fonction `forever` ce qui implique, que nous allons répéter indéfiniment le code suivant. Les trois premiers `if(...)` permettent de savoir si nous avons reçu des données de la part des émetteurs (grâce à la fonction `availableDataFromName`), si tel est le cas, alors on enregistre cette donnée dans la variable associée (en utilisant `readDataFromName`).

Maintenant que nous avons les données, il faut les afficher sur notre écran de LEDs, c'est le rôle de la dernière condition `if`, qui nous permet de mettre à jour notre écran avec les nouvelles données, seulement si cela fait plus de deux seconds qu'il n'a pas été rafraîchi.

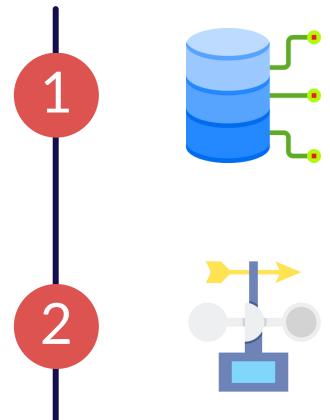
MAGNETICS - CRÉER DES PROJETS MULTICARTES



ETAPE 3 - AMÉLIORER

Cette activité ne fait qu'afficher les informations sur un écran, mais il est possible d'utiliser le **datalogger**, pour enregistrer nos données et les analyser.

Utiliser une boîte étanche et faire une **station météo**, la connexion étant sans fil, on pourrait avoir un écran en intérieur nous affichant les données environnementales de l'extérieur.



ALLER PLUS LOIN

Bluetooth - Découvrir plus d'informations sur cette norme de télécommunication si présente dans notre vie quotidienne

<https://fr.wikipedia.org/wiki/Bluetooth>



CircuitPython BLE Libraries on Any Computer - Utilisez le code BLE de CircuitPython sur les ordinateurs de bureau, les ordinateurs portables et les Raspberry Pi grâce aux bibliothèques Adafruit

<https://learn.adafruit.com/circuitpython-ble-libraries-on-any-computer>



Bouton de volume BLE avec CircuitPython - Transformez votre Bluefruit de Circuit Playground en bouton de volume BLE sans fil

<https://learn.adafruit.com/bluetooth-le-hid-volume-knob-with-circuitpython>



partie 3

S'inspirer - Idées de projet

magnetics – un projet inscrit dans un maillage d'initiatives

au service de la pensée computationnelle et de la créativité

Le projet magnetics s'inscrit dans un réservoir d'initiatives, interconnectées et menées en parallèle par un consortium de partenaires européens, dont le L.A.B est garant du volet technique. Toutes ces initiatives ont pour objectif de participer à l'enrichissement de modalités pédagogiques pluridisciplinaires s'appuyant sur des outils numériques comme vecteur d'apprentissage. L'objectif de cette approche multi-dimensionnelle (à la fois sur les contenus, la stratégie, la pédagogie, la technique et les outils numériques) est de travailler à réintroduire chez les élèves, le goût et la motivation pour les STEAM grâce au développement de pédagogies actives, proposant à l'élève une approche par projet, par l'expérimentation et basées sur 5 compétences clés du 21ième siècle (Romero, M. 2017) : la créativité, la collaboration, la pensée critique, la résolution de problème et la pensée computationnelle.

Au travers de chacun de ces projets, nous avons produit ou produisons actuellement plusieurs propositions de projets ou de protocoles scientifiques permettant d'aborder des défis sociétaux, porteurs de sens pour les élèves, grâce à la programmation. Notre ambition est de développer le sentiment de maîtrise d'un sujet, en outillant les enseignants et les élèves avec des ressources numériques concrètes.

La brique magnetics est utilisée dans chacun de ces projets pour créer des activités de plus grande envergure, notamment au sein de territoires étendus.



Le projet Let's STEAM

En 2016, STMicroelectronics en partenariat avec AMU et le L.A.B ont lancé l'initiative STM32 pour l'éducation, visant à rendre les microcontrôleurs produits par ST utilisables à l'école. Durant ce projet, nous avons rassemblé une trentaine d'enseignants afin de définir leurs besoins et essayer de trouver des solutions adaptées aux enjeux de la classe. En quelques mois, nous nous sommes aperçus que les enseignants non technophiles avaient quitté l'initiative, car ils ne se sentaient pas légitimes face à ces outils malgré leur potentiel. Fort de cette observation, AMU et le L.A.B ont contacté plusieurs de leurs partenaires historiques ainsi que des acteurs de la scène européenne des STEAM, afin de travailler ensemble à des contenus de formation pour les enseignants adaptés à tous dans l'optique de stimuler la créativité et d'aborder les enjeux liés à la pensée computationnelle. La réunion de ces acteurs a permis de créer l'initiative Let's STEAM, rassemblant des visions complémentaires entre le monde de la recherche, de l'électronique, de la programmation et de l'enseignement en milieu scolaire.

A travers la mise en œuvre de Let's STEAM, nous souhaitions apporter de nouvelles compétences ou approfondir les connaissances des enseignants dans le domaine des technologies techno créatives, afin de promouvoir des approches STEAM dynamiques et engageantes pour les professeurs comme pour les élèves. Grâce à une approche transdisciplinaire de l'utilisation des cartes programmables, le programme de formation Let's STEAM souhaitait promouvoir des pédagogies actives et innovantes, en particulier les approches par enquête et les démarches expérimentales. Pour soutenir ces objectifs, un enjeu du projet était de développer des outils au service de la programmation porteuse de sens, adaptés à l'environnement de la classe et aux contraintes des enseignants. L'ambition du projet était d'avoir une approche inclusive (notamment pour les élèves éloignés des sciences ou de la technologie e.g. les jeunes filles) en abaissant la barrière à l'entrée des technologies issues de la microélectronique européenne, au service de l'engagement éducatif.

Afin de répondre à ces objectifs, les partenaires du projet Let's STEAM ont défini un programme de formation articulé autour de 3 thèmes : la programmation au service de la créativité, les démarches par enquêtes et la création d'activités technocreatives inclusives. La totalité du programme a été consolidée sous forme d'un manuel de cours, utilisable pour former les professeurs, ou directement comme ressources documentaires dans la classe.

Ce manuel est accessible ici :

https://github.com/letssteam/Resources/blob/main/1_Full_Coursebook/Lets_STEAM_Coursebook_FR.pdf

En savoir plus : www.lets-steam.eu



Le projet TheDexterLab

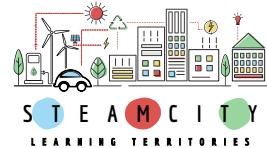
Le projet TheDexterLab vise à développer des solutions simples et abordables pour maintenir des activités d'expérimentation scientifique à l'intérieur et à l'extérieur de la classe en stimulant la créativité des élèves et l'utilisation d'outils numériques. L'accès à l'instrumentation scientifique dans l'enseignement secondaire est souvent entravé par l'inégale disponibilité des ressources dans le monde de l'éducation entre les pays et les écoles d'Europe. Cela conduit à une faible diversité des outils d'expérimentation disponibles, compliquant la possibilité pour les enseignants de développer des activités STEAM (Science, Technologie, Ingénierie, Arts, Mathématiques) à grande échelle, dans une logique interdisciplinaire et axée sur une meilleure compréhension par les élèves des défis sociétaux de notre siècle, illustrant le lien entre les questions scientifiques et techniques et leur impact sur la vie quotidienne des citoyens.

Par ailleurs, la crise actuelle du COVID-19 a accentué les inégalités entre les étudiants en raison de l'hybridation nécessaire mais complexe des modalités d'apprentissage. Pendant la pandémie, les activités expérimentales ont donc été lourdement affectées. Elles restent pourtant cruciales pour comprendre les processus scientifiques. Le projet TheDexterLab a été conçu pour répondre à ce défi à court et à long terme en créant plusieurs ressources associées à des protocoles d'expérimentation scientifique, permettant de maintenir et de stimuler l'utilisation de pédagogies actives dans l'enseignement secondaire. Ces ressources comprennent :

- Le développement d'instruments scientifiques de collecte et d'analyse de données, ou dispositifs DIY, accessibles à tous à faible coût, modulaires et réutilisables, conçus à travers une approche Do-It-Yourself pour être facilement reproductibles à l'école, au sein de fablabs ou à la maison.
- Le développement d'outils de simulation, permettant de pallier le manque de ressources financières et de maintenir les activités scientifiques sans dégradation, que le matériel soit disponible pour l'élève/la classe, avec l'objectif sous-jacent de stimuler les compétences en programmation numérique des apprenants.
- Enfin, une documentation de soutien et des lignes directrices permettant la bonne compréhension et la mise en œuvre des protocoles TheDexterLab dans des conditions idéales, au sein d'écosystèmes d'apprentissage formels et informels.

Pour en savoir plus : www.thedexterlab.eu

Le projet SteamCity



Grâce au projet Let's STEAM, le L.A.B a lancé sa première initiative en tant que coordinateur d'un projet Erasmus + de coopération : STEAMCITY. Le projet SteamCity vise à développer et offrir un ensemble de services d'expérimentation interdépendants dédiés à la mise en œuvre d'enquêtes STEAM dans l'enseignement secondaire, afin d'aborder les thèmes des territoires intelligents et apprenants. Le projet a pour ambition de promouvoir les écoles en tant qu'incubateurs de projets de science citoyenne, en donnant aux élèves les moyens de mobiliser leurs connaissances interdisciplinaires dans le cadre de défis sociaux, sociaux et environnementaux. En tant que tel, le projet SteamCity contribue à des axes majeurs des politiques éducatives actuelles, justifiant cette demande de financement :

- Développer des expériences d'apprentissage scientifique stimulantes : Grâce à la démarche scientifique, l'être humain dispose des outils intellectuels pour devenir un acteur conscient et responsable dans son rapport au monde et dans la transformation des sociétés. Dans ce cadre, l'ambition de SteamCity est d'aider les écoles, les enseignants et les élèves à adopter une approche de l'enseignement scientifique basée sur l'investigation, en encourageant le développement d'expériences d'apprentissage significatives et stimulantes, visant à développer l'esprit critique.
- Offrir des expériences STEAM inclusives : Développer de meilleures expériences d'apprentissage pour les étudiants signifie promouvoir l'inclusion. Le défi de l'égalité dans l'enseignement des STEM est un sujet crucial car les domaines scientifiques véhiculent encore des stéréotypes de genre persistants, en plus des inégalités dans l'accès aux ressources nécessaires à la promotion des pratiques d'apprentissage par la pratique. Participer à l'émergence de modèles et d'environnements éducatifs plus inclusifs et significatifs est donc un objectif de SteamCity.

Le projet s'appuie sur la création d'un modèle d'enquête promouvant l'interdisciplinarité et pouvant être instancié au travers de diverses ressources inspirantes ainsi que le développement d'outils d'expérimentation, s'appuyant sur des pratiques de programmation et des dispositifs de collecte de données.

Pour en savoir plus : www.steamcity.eu

Les inspirations et ressources

Les tableaux ci-dessous vous donnent accès à différentes idées de projet que vous pourriez mettre en œuvre grâce à nos différentes ressources ! N'hésitez pas à vous emparer de chaque proposition et à utiliser magnetics pour les transformer en expérimentation porteuse de sens et motivante !

Idées de projet Let's STEAM

Chaque projet Let's STEAM a été décrit plus en détails au sein d'un wiki dédié. Vous pouvez accéder à chaque projet en visitant la page suivante : <https://bit.ly/3Oc62eF>

NOM DU PROJET	OBJECTIFS
Comment rendre visible l'invisible ?	Reproduire l'environnement naturel des grenouilles afin d'assurer leur survie
Préserver la biodiversité	Surveillez le nombre d'espèces végétales dans votre quartier. Explorez les rues et les parcs de votre quartier pour en savoir plus sur l'écosystème et utilisez la technologie pour faciliter ce processus !
Contrôle de la température dans la classe	Il fait trop chaud dans la salle de classe. Lorsque les élèves entrent, ils savent qu'ils doivent fermer les stores, mais pendant la pause, la classe devient vraiment chaude. Comment créer un système plus autonome grâce à la programmation ?
Construire une salle de classe accueillante	Identifiez les besoins en intensité lumineuse particulière dans votre classe pour réaliser une activité spécifique.
Votre maison idéale (et durable)	Rêvez à l'endroit où vous aimeriez vivre, à ce que serait votre maison idéale et à la façon dont cette maison idéale pourrait être plus durable.
Gestes barrières	Bien que de nouvelles routines aient été mises en place pour s'assurer que tous les enfants se lavent les mains, comment la programmation peut nous aider à respecter les gestes barrières ?
Consommation raisonnable de chauffage	Identifiez la position optimale d'utilisation des appareils de chauffage, à des moments donnés, pour économiser l'électricité.

Les protocoles d'expérimentation TheDexterLab

Chaque protocole TheDexterLab fait l'objet d'une fiche d'activité spécifique accessible au sein d'un wiki dédié. Vous pouvez accéder à chaque protocole en visitant la page suivante : <https://bit.ly/3tAkrb9>

PROTOCOLE	DÉFI ABORDÉ
Pourquoi l'océan est-il salé ?	Comment séparer le sel et l'eau et contrôler le résultat ?
Le son se déplace-t-il de plus de 100 mètres en 1 seconde ?	A quelle vitesse le son voyage-t-il ?
Quand faut-il arroser une plante ?	Comment irriguer automatiquement une plante ?
Votre temps de réaction est-il inférieur à une demi-seconde ?	Comment évaluer le temps de réaction d'un cycliste ?
Est-ce que notre corps ou ses parties (par exemple les mains, les jambes, etc.) sont accélérés de plus de 1 g même si nous dansons comme des fous ?	Quelle force notre corps ou ses parties subissent-ils lorsque nous bougeons, courons ou dansons ?
La distraction peut-elle modifier votre temps de réaction ?	Comment évaluer le temps de réaction d'un cycliste ?
Le vent et les chutes d'eau ont-ils de l'énergie ?	Comment pouvons-nous récolter de l'énergie pour produire de l'électricité à partir de sources renouvelables ?
La lumière a-t-elle de l'énergie ?	Comment pouvons-nous récolter de l'énergie pour produire de l'électricité à partir de sources renouvelables ?
La qualité de l'air diminue-t-elle lorsque le nombre de personnes présentes dans une pièce augmente ?	Comment améliorer la qualité de l'air en milieu fermé ?
Peut-on mesurer l'atténuation du son par un matériau ?	Quels sont les matériaux habituels qui peuvent atténuer le son ?
Peut-on construire une machine qui utilise le vent pour soulever un poids de 50 g ?	Peut-on utiliser le vent comme source d'énergie mécanique ?
Comment maximiser l'apport en énergie solaire et créer des panneaux solaires auto-orientables ?	Comment construire des systèmes solaires efficaces ?

Les protocoles TheDexterLab – suite

PROTOCOLE	DÉFI ABORDÉ
Comment pouvons-nous réduire la quantité d'énergie que nous utilisons ?	Comment faire des économies d'énergie ?
Une plante consomme-t-elle plus de CO ₂ qu'elle n'en rejette ?	Comment améliorer la qualité de l'air ?
L'IA peut-elle être un outil de sécurité sans faille ?	Comment détecter les visages ?
La température modifie-t-elle la vitesse du son ?	Est-il possible de mesurer la variation de la vitesse du son en fonction de la température ambiante ?
Commander les équipements en fonction de l'énergie produite par le panneau solaire	Identifier différentes ressources en énergie et connaître quelques conversions d'énergie

Les enquêtes SteamCity sur la ville intelligente

Les enquêtes SteamCity sont en cours de développement. Dès qu'ils seront finalisés (d'ici Mai 2024), ils seront mis à disposition gratuitement sur le site www.steamcity.eu.

THÈME ABORDÉ	QUESTIONS ET DÉFIS	IDÉES D'EXPÉRIMENTATION
Pollution lumineuse nocturne	Quel est l'impact de la pollution lumineuse dans les zones urbanisées sur la qualité du sommeil ?	Créer une expérience pour comparer les niveaux de pollution lumineuse à la fenêtre des chambres des élèves et évaluer l'impact sur la qualité du sommeil selon cette cartographie
Qualité de l'air	Quel est l'impact de la météo sur les particules ? Quel est l'impact des cosmétiques ?	Mesurer les particules en fonction de la vitesse du véhicule et de plusieurs niveaux de hauteur par rapport aux émissions
Réglementation de la mobilité	Comment les différents modes de mobilité se partagent-ils la ville ?	Créer un détecteur qui vérifie si un véhicule se trouve dans l'espace qui lui est réservé. Créer un contrôle de vitesse pédagogique et en mesurer l'impact. Lumière de circulation synchronisée pour réduire la pollution

Les enquêtes SteamCity - suite

THÈME ABORDÉ	QUESTIONS ET DÉFIS	IDÉES D'EXPÉRIMENTATION
Impact environnemental de la mobilité	Quel est l'impact environnemental de la vitesse des véhicules ? La mobilité collective réduit-elle son impact environnemental ?	Créer un lecteur de limite de vitesse avec l'apprentissage automatique
Pollution sonore	Quel est l'impact des nuisances sonores en milieu urbanisé sur le niveau de stress et la santé des êtres humains ?	Créer un modèle et une expérience pour questionner le lien entre le bruit et les capacités d'apprentissage Créer une cartographie sensible des nuisances sonores dans une ville ou dans un quartier Comprendre comment fonctionne l'atténuation du bruit par les différents matériaux qui composent la ville
Lien entre la faune et la flore dans la biodiversité	Quel est l'impact de l'action humaine sur la biodiversité ? Peut-on enrichir la biodiversité en transformant nos paysages ? Quel est le degré de fragilité des écosystèmes ?	Créer une cartographie sensible des pollinisateurs dans la ville Créer une cartographie des espèces d'oiseaux grâce à des camera traps Créer un terrarium Créer une cartographie sensible des liens entre faune et flore
Lumière et biodiversité	Comment une ville intelligente peut-elle mieux gérer l'éclairage public ? Quel est l'impact de la pollution lumineuse sur la biodiversité dans les zones urbanisées ?	Installer un détecteur sur une saison pour voir la saison de ponte des tortues de mer Comparer en classe les plantes exposées à la lumière artificielle et regardez les différences entre nous.
Processus d'apprentissage artificiel bio-inspiré	Comment les gens utilisent-ils leurs capacités d'apprentissage pour s'adapter au monde de demain ? Comment l'être humain adapte-t-il ses comportements ?	Faites l'expérience des processus d'apprentissage collaboratif en jouant Faire un robot suiveur de ligne simple Optimiser le comportement du suiveur de ligne par l'apprentissage par renforcement et le faire traverser un diorama de ville.

Les enquêtes SteamCity - suite

THÈME ABORDÉ	QUESTIONS ET DÉFIS	IDÉES D'EXPÉRIMENTATION
Classification des images	Comment reconduire un concept avec seulement un ensemble limité d'images ?	Classification des déchets grâce à l'IA
Éthique de l'IA	Quel est l'impact de l'IA sur la vie quotidienne ? L'IA est-elle objective et libre de tout préjugé humain ?	Comment créer un ensemble de données sans nos propres préjugés invisibles ?
Température et climat	Quel est l'impact du changement climatique ?	Créer des expériences pour comprendre l'impact des températures sur les être vivants aquatiques ou terrestres

partie 4

Aller plus loin

**Découvrir les technologies
derrière magnetics et
l'architecture du prototype**

BLE Mesh

Qu'est-ce que le BLE Mesh et comment cela fonctionne-t-il ?

Le Bluetooth Low Energy Mesh est une technologie de maillage qui permet entre autres choses, une extension de la portée de communication entre objets d'un même réseau et aussi une transmission sans fil bidirectionnelle.

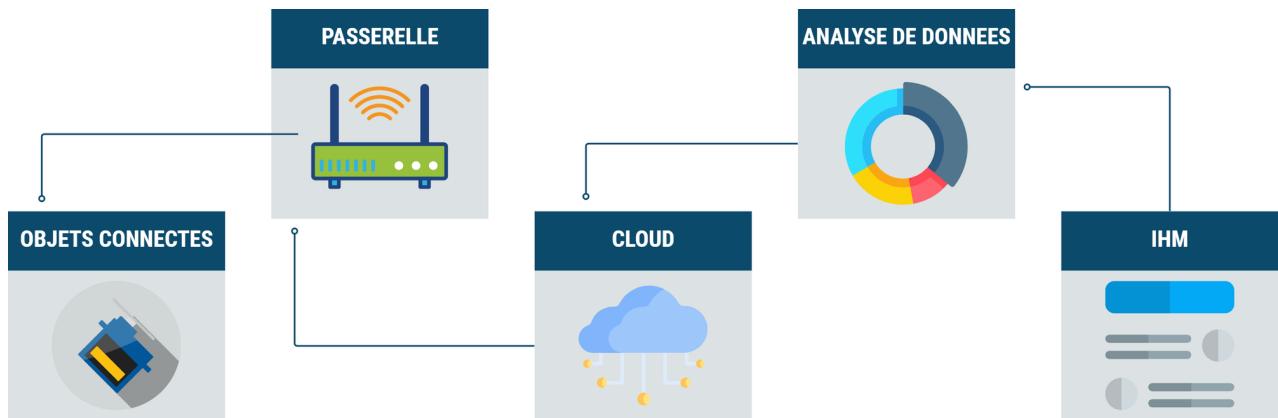
La solution contient la fonctionnalité de base pour une communication sécurisée et offre toute la flexibilité nécessaire pour créer des applications.

Bluetooth Low Energy (BLE) est devenu la norme préférée pour la connectivité IoT. Le fonctionnement inhérent à faible consommation de la norme BLE, associé à la prise en charge installée sur les smartphones et les ordinateurs personnels, a été un moteur de sa croissance des applications IoT.

Les applications BLE traditionnelles utilisent la communication point à point, où chaque paire d'appareils envoie des données entre eux. Chacune de ces connexions utilise soit le rôle GAP Central, soit le rôle GAP Peripheral pour réaliser la communication.

Introduction au BLE Mesh et IoT

La technologie Bluetooth Low Energy (BLE) Mesh connecte plusieurs appareils avec un réseau maillé adapté à l'Internet des objets (IoT).



Le BLE Mesh peut être utilisé dans plusieurs applications nécessitant un transfert de données peu fréquent pour créer des systèmes de contrôle distribués tels que:

- Éclairage intelligent
- Domotique
- L'automatisation industrielle

Dans un réseau Mesh, chaque appareil peut communiquer avec tous les autres appareils du même réseau. Cette capacité étend la portée de communication globale au-delà de la portée de chaque appareil individuel.

Avec le Bluetooth Mesh, les messages sont envoyés à l'aide de paquets dit advertising. En utilisant ces paquets d'advertising pour la communication, le relais des messages ne nécessite pas de connexion entre les différents nœuds du réseau.

Les données sont diffusées par un émetteur (en utilisant ces paquets d'advertising) et sont reçues par tous les dispositifs qui se trouvent à portée. Les périphériques qui sont inclus dans un réseau BLE Mesh sont appelés simplement «nœuds».

Le Bluetooth Mesh définit différents types de nœuds; certains sont utilisés pour l'extension de la portée et la couverture du réseau tandis que d'autres sont optimisés pour un fonctionnement à faible puissance et ne se réveillent que lorsqu'ils en ont besoin. Le BLE Mesh est en train de devenir le meilleur choix technologique pour la domotique car les nœuds BLE Mesh peuvent être contrôlés directement à partir d'un téléphone mobile ou d'un PC sans avoir besoin d'installer une passerelle réseau additionnelle. La spécification BLE Mesh a été définie pour fournir des modèles d'exploitation normalisés adaptés aux cas d'utilisation basique. Les formats de message standard pour des cas d'utilisation définis (modèles) permettent un déploiement rapide et une garantie d'interopérabilité avec les autres produits BLE Mesh.



Typologie et fonctionnalités des noeuds BLE Mesh

Le BLE Mesh a plusieurs types de nœuds. Un type de nœud est associé à un ensemble de fonctionnalités. Chaque nœud pourra être associé à un ou plusieurs type de nœud en fonction de l'usage qu'il est prévu d'en faire.

Nœud de relais (Relay Node)

La fonctionnalité de relais d'un nœud comprend la possibilité de relayer des messages d'advertising. Dans un réseau maillé BLE Mesh, la plupart des nœuds ont cette caractéristique s'ils ont accès à une source d'énergie. Par exemple une ampoule intelligente ou un interrupteur d'éclairage disposeront de cette caractéristique. En règle générale, tout nœud alimenté est susceptible d'être un nœud de relais, car il dispose de la puissance nécessaire pour écouter en permanence les paquets d'advertising et les relayer vers les autres nœuds du réseau.

Nœud basse consommation (Low Power Node ou LPN)

La fonctionnalité nœud à faible consommation d'énergie (Low Power Node ou LPN) est essentielle pour les applications compatibles BLE Mesh. Les LPN fonctionnent dans des cycles d'utilisation relativement larges, en travaillant et en communiquant en conjonction avec des nœuds amis (Friend) qui collectent des messages au nom du LPN.

Le LPN enverra une requête ping à son nœud ami à un intervalle défini pour vérifier les messages en attente, et après la communication avec le nœud ami, il retournera à un état de veille à faible consommation pour économiser son énergie. Toute application alimentée sur batterie, telle qu'un capteur d'ouverture de porte ou un détecteur de fumée, sera généralement un LPN. Avec la fonctionnalité LPN du BLE Mesh, une simple pile bouton est suffisante pour alimenter un nœud pendant plus d'un an.

Nœud ami (Friend Node)

Un nœud doté de la fonction Friend écoutera tous les messages relayés sur le réseau et destinés à ses LPN associés. Le nœud Friend stockera ces messages et les remettra aux LPN associés lorsque les LPN se réveillent et l'interrogent.

Étant donné qu'un nœud ami doit stocker des messages pour un ou plusieurs LPN, le nœud ami peut nécessiter plus de mémoire que les autres types de nœud. La quantité de mémoire requise dépend de la quantité de données / commandes devant être stockées sur le nœud Friend qui seront communiquées au LPN pendant une opération d'interrogation.

Nœud proxy (Proxy Node)

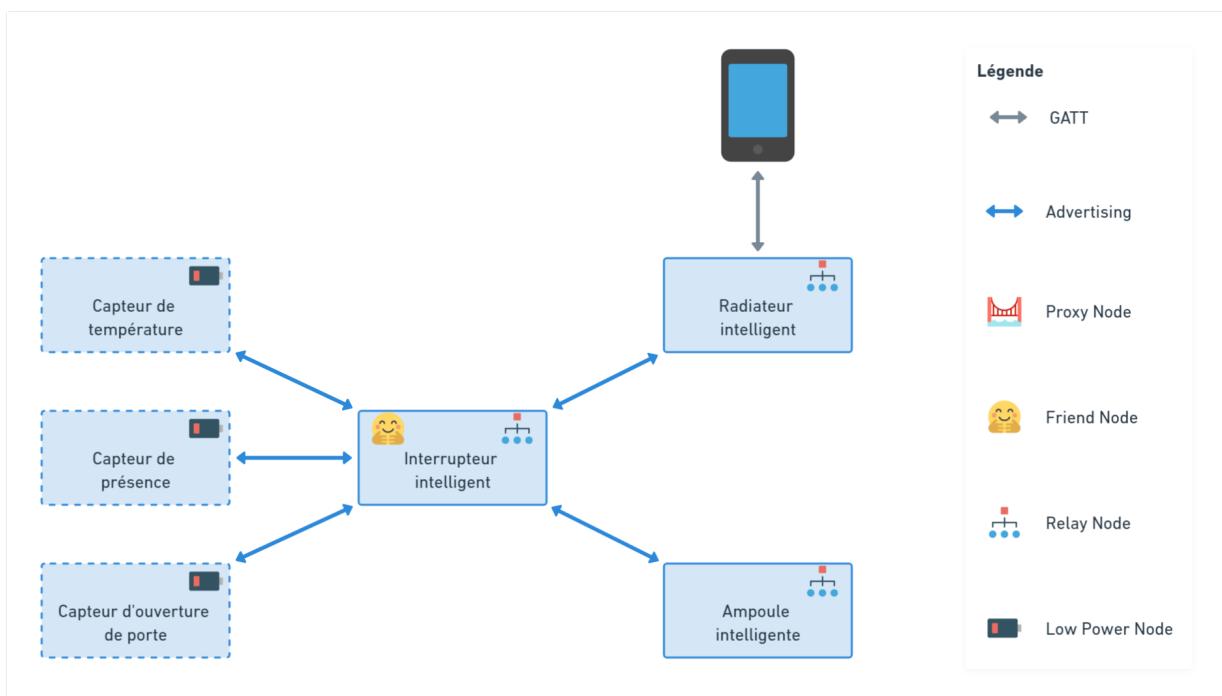
Les réseaux BLE Mesh relaient les messages en utilisant les paquets d'advertising du BLE. La fonction Proxy est la capacité d'un nœud à relayer des messages entre le protocole GATT (General ATTribute) du BLE et les paquets d'advertising du BLE Mesh. Cette fonctionnalité permet aux appareils, tels que les smartphones, qui prennent en charge BLE mais pas BLE Mesh, de communiquer avec un réseau Mesh.

Un nœud proxy est le point d'entrée d'un réseau Mesh pour des périphériques qui ne prennent pas directement en charge le BLE Mesh. Tout nœud prenant en charge la fonction Proxy peut servir d'interface pour un smartphone / PC via une connexion BLE classique et l'utilisation du protocole GATT.

Nœud d'approvisionnement (Provisioner Node)

La fonction d'approvisionnement est la clé pour activer un réseau maillé BLE sécurisé. L'approvisionnement est le processus d'ajout d'un nouveau nœud à un réseau existant. Lorsqu'un client achète un appareil compatible BLE Mesh, il doit avoir la possibilité d'ajouter cet appareil à son réseau. Cela nécessite plusieurs étapes pour garantir que des périphériques non désirés ne puissent pas être provisionnés sur le réseau.

Un appareil non provisionné enverra des messages à un intervalle prédéterminé et le nœud d'approvisionnement lancera le processus une fois l'appareil trouvé et sélectionné. Le protocole BLE Mesh utilise l'algorithme ECDH (Elliptic Curve Diffie Hellman) pour l'échange de clés sécurisée. Cela permet de garantir la sécurité de votre réseau et de l'approvisionnement des appareils.



Fonctionnement et concepts clés du BLE Mesh

Managed flood

Le BLE Mesh utilise une le principe de l'inondation gérée (Managed flood) pour transférer des messages entre les nœuds du réseau. L'inondation gérée est un mode de communication à chemins multiples qui ajoute suffisamment de redondance pour garantir qu'un message atteigne sa destination. Dans une implémentation naïve où chaque nœud relaie aveuglément tous les messages qu'il a reçu à ses voisins sans se préoccuper de savoir si le message a déjà été transmis, le réseau maillé serait inondé de message qui se répéterait à l'infini. À l'inverse, l'opération d'inondation gérée du BLE est conçue pour empêcher les périphériques de relayer les messages déjà reçus précédemment en ajoutant tous les messages dans une liste mise en cache. Lorsqu'un message est reçu, il est comparé avec les éléments de cette liste et ignoré s'il est déjà présent. Si le message n'a pas été reçu auparavant, il est ajouté au cache afin qu'il puisse être ignoré à l'avenir. Chaque message comprend une valeur de durée de vie (TTL) qui limite le nombre de fois qu'un message peut être relayé.

Communication basée sur la publication et abonnement (Publish/Subscribe)

Un nœud "Publisher" envoie des messages et seuls les nœuds qui se sont abonnés recevront ces messages. Cela garantit à plusieurs types de produits de coexister dans un même réseau sans être dérangés par les messages inutiles. Par exemple, dans une maison, les ampoules de chaque pièce s'abonneront uniquement aux interrupteurs de cette même pièce. Une ampoule n'a pas besoin d'agir quand un interrupteur de la pièce d'à côté change d'état. De plus, les messages peuvent être soit en "monodiffusion", soit en "multidiffusion" et/ou en diffusion large, ce qui signifie qu'un message peut atteindre un seul, quelques-uns ou tous les nœuds du réseau à la fois.

Éléments

Les éléments sont des entités qui définissent la fonctionnalité d'un nœud. Chaque nœud peut avoir un ou plusieurs éléments. Chaque nœud a au moins un élément appelé «élément primaire». Par exemple, une ampoule à intensité variable a généralement un élément; cet élément peut exposer une ou plusieurs fonctionnalités telles que "ON/OFF" et le contrôle de niveau de luminosité. Dans cet exemple, le modèle de serveur Light Lightness est utilisé pour obtenir la fonctionnalité ON/OFF et de contrôle de niveau de luminosité.

Un autre exemple est une ampoule à intensité variable qui comprend également un capteur de présence. Dans cet exemple, le nœud aura deux éléments: un pour la fonction d'éclairage et l'autre pour la fonction capteur. L'élément principal dans ce cas serait la fonction d'éclairage, et le modèle de serveur Light Lightness pour les commandes d'éclairage. L'élément secondaire est le capteur et l'états seront accessibles à travers cet élément.

Chaque élément d'un nœud a une adresse unique, connue sous le nom d'adresse de monodiffusion. Cela permet à chaque élément d'être adressé indépendamment.

Modèles

Les modèles sont utilisés pour définir la fonctionnalité d'un nœud - ils rassemblent les concepts d'états, de transitions, de liaisons et de messages pour un élément. Les modèles sont analogues aux services du Bluetooth. Il existe trois types de modèles de maillage: les modèles de client, les modèles de serveur et les modèles de contrôle (qui implémentent à la fois un client et un serveur dans un seul modèle).

- Modèle de serveur - Un modèle de serveur peut avoir un ou plusieurs états couvrant un ou plusieurs éléments. Le modèle de serveur définit les messages qu'un nœud implémentant ce modèle peut transmettre et recevoir. Il définit également les comportements de l'élément en fonction de ces messages. Autrement dit, les modèles de serveur exposent les états de l'élément qui peuvent être lus ou contrôlés par un client. Un exemple d'application du modèle de serveur est un capteur qui expose l'état du capteur ou une ampoule qui stocke l'état actuel de la lumière.
- Modèle de client - Un modèle de client définit l'ensemble de messages pour demander et modifier l'état d'un serveur. Par exemple, un interrupteur ON/OFF (fonctionnant en tant que client) peut envoyer un message à un serveur ON/OFF pour changer l'état d'un appareil (ampoule ou radiateur par exemple).
- Modèle de contrôle - Une application finale peut utiliser des modèles de serveur ou des modèles de client, ou les deux avec la logique de contrôle. Toute combinaison de modèles serveur et client aboutit à un modèle de contrôle.

Hiérarchie de modèles

Les modèles peuvent étendre les fonctionnalités d'autres modèles. Autrement dit, les modèles peuvent être hiérarchiques. Par exemple, le modèle «Light Lightness» étend le modèle «Generic OnOff Server» et le modèle «Generic Level Server». Cela signifie que si vous implementez le modèle Light Lightness dans une application, vous obtenez toutes les fonctionnalités Light Lightness ainsi que toutes les fonctionnalités Generic Level et Generic OnOff. Les modèles qui ne s'étendent pas à d'autres modèles sont appelés «modèles racines».

États et propriétés

- États : Les éléments peuvent être dans diverses conditions; dans Bluetooth Mesh, ces conditions sont stockées dans des valeurs appelées états. Chaque état est une valeur d'un certain type contenue dans un élément. En plus des valeurs, les états ont des comportements associés à cet état. Les états sont définis par le SIG Bluetooth. Par exemple, il existe un état appelé «Generic OnOff» qui peut avoir deux valeurs - ON ou OFF. Ceci est utile pour les appareils tels que les ampoules ou les moteurs de ventilateurs. Le terme «Générique» est utilisé pour indiquer que cet état et ses comportements peuvent être utiles dans différents types de périphériques Mesh.
- Propriétés : Les propriétés contiennent également des valeurs relatives à un élément; cependant, contrairement aux états, les propriétés fournissent le contexte pour interpréter les états. Par exemple, considérons un appareil qui souhaite envoyer une valeur d'état de température. L'état de température peut être «Température ambiante intérieure actuelle» ou «Température ambiante extérieure actuelle». Dans ce cas, une propriété serait utilisée pour fournir le contexte de la valeur de l'état de température. Les propriétés peuvent être des propriétés Manufacturer, qui sont en lecture seule ou des propriétés dite Admin, qui permettent un accès en lecture-écriture.
- Transitions d'État : Les transitions d'état peuvent être instantanées ou s'exécuter sur une période appelée temps de transition.
- Liaison d'État : Les États peuvent être liés entre eux de telle sorte qu'un changement dans un état entraîne un changement dans l'autre. Un état peut être lié à plusieurs autres états. Par exemple, une lumière contrôlée par un gradateur aura un état appelé Generic OnOff et un autre appelé Generic Level pour spécifier la luminosité. Si la lumière est à l'état ON mais est atténuée au point que le niveau passe à zéro, l'état OnOff passera à OFF. La liaison d'état est définie par les modèles contenant les états en question. Ceux-ci peuvent être trouvés dans la spécification Bluetooth Mesh .

Sécurité et confidentialité

La sécurité et la confidentialité sont toujours une préoccupation pour les appareils connectés sans fil. Le BLE Mesh intègre plusieurs techniques pour atténuer ce problème. Le tableau ci-après détaille les méthodes de sécurité et de confidentialité implémentées dans une solution BLE Mesh.

Chiffrement et authentification	Tous les messages Bluetooth Mesh sont cryptés et authentifiés.
Séparation des préoccupations	La sécurité du réseau, la sécurité des applications et la sécurité des appareils sont traitées indépendamment au moyen de clés distinctes. Une AppKey est utilisée pour éviter tout conflit d'intérêts et différencier un type d'application d'un autre. Par exemple, un nœud de capteur ne peut pas décoder un message chiffré avec la clé AppKey d'éclairage. Chaque nœud possède une ou plusieurs NetKeys en fonction du réseau et des sous-réseaux dont ils font partie. La NetKey sépare chaque réseau et sous-réseau de tous les autres. Chaque nœud a une DevKey unique qui est utilisée pour l'approvisionnement et la configuration qui le sépare de tous les autres périphériques.
Isolation de zone	Un réseau Bluetooth Mesh peut être divisé en sous-réseaux, chacun étant cryptographiquement distinct et sécurisé des autres. Par exemple, chaque chambre d'un hôtel peut être un sous-réseau tandis que l'hôtel est le réseau complet. Les sous-réseaux permettent aux clients d'une pièce de ne pas interférer avec les autres pièces.
Actualisation des clés, protection contre les attaques de la corbeille et suppression des nœuds	Les clés de sécurité peuvent être modifiées pendant la durée de vie du réseau maillé Bluetooth via une procédure d'actualisation des clés. Lorsqu'un nœud est supprimé du réseau, le réseau maillé exécute la procédure d'actualisation des clés pour modifier les clés sur tous les autres nœuds. Une fois les clés actualisées, les (anciennes) clés stockées sur le nœud qui ont été supprimées n'ont aucune valeur. Cette fonctionnalité résout les problèmes de sécurité si quelqu'un accède à un périphérique qui était présent auparavant dans un réseau BLE.

Sécurité et confidentialité - suite

Replay Attack Protection	La sécurité du Bluetooth Mesh protège le réseau contre les attaques de relecture. Les attaques de relecture sont l'une des attaques les plus courantes dans lesquelles un espion écoute un message puis le rejoue avec une mauvaise intention. Imaginez que quelqu'un écoute un message de déverrouillage de porte, puis le rejoue au milieu de la nuit pour entrer par effraction dans votre maison. Le BLE Mesh fournit un mécanisme pour éviter les attaques de relecture en ajoutant un numéro de séquence dans chaque message. Les messages avec le même numéro de séquence ou un numéro de séquence inférieur à un message précédent sont automatiquement ignorés.
Message Obfuscation	Le masquage des messages rend difficile le suivi des messages envoyés au sein du réseau et, en tant que tel, fournit un mécanisme de confidentialité pour rendre difficile le suivi des nœuds.
Provisionnement sécurisé des appareils	Le BLE Mesh fournit une méthode sécurisée pour ajouter un appareil au réseau.



Prototype magnetics

Principes, architecture et protocoles

Le projet MAGNETICS a pour objectif technique de développer un prototype maillé entre Scratch, MakeCode et CircuitPython, afin de réutiliser et capitaliser les données collectées par plusieurs cartes électroniques et exploitées sur plusieurs systèmes au sein de différentes disciplines du spectre STEAM, en stimulant l'interdisciplinarité et en permettant d'optimiser l'utilisation des outils à disposition des enseignants.

Le projet MAGNETICS est un ensemble de trois briques techniques :

- bluetooth low energy : Technologie de maillage BLE permettant de connecter et de créer un dialogue entre un plus grand nombre de dispositifs, en étendant la portée de la communication.
- interopérabilité : garantissant la compatibilité entre les cartes de programmation, notamment les principales solutions éducatives du marché, à savoir STM32, Raspberry Pi et micro:bit.
- maillage numérique : entre Scratch, MakeCode et CircuitPython (principaux outils d'apprentissage de la programmation utilisés dans les écoles secondaires).

Techniquement, l'interopérabilité sera obtenue grâce à une pile de protocoles dynamique adaptée aux contraintes du MCU embarqué à bas coût. Ce travail est mené conjointement avec l'équipe RiSE de Microsoft Research.



Principe

L'objectif de la solution magnetics est de permettre à tout enseignant sans compétence particulière en développement logiciel ou en électronique de créer une solution IoT pour ses besoins pédagogiques. L'équipement doit pouvoir oublier au maximum les aspects techniques pour permettre à l'utilisateur de se concentrer sur les usages qu'il veut démontrer . Simplicité, minimalisme et automatisation maximale sont donc des qualités nécessaires.

Dans la phase de création de la preuve de concept, l'approche la plus efficace pour converger reste l'approche expérimentale avec ses essais/erreurs consécutifs. Le prototype doit pouvoir changer et évoluer très fréquemment sans remettre en cause ce qui a déjà été mis en place. La modularité et l'indépendance des éléments matériels seront donc également un aspect central.

Dans les phases initiales d'un projet, la sécurité est souvent mise de côté. Pourtant, dès qu'une preuve de concept est présentée, elle peut faire l'objet d'une attaque extérieure. Il est donc essentiel que notre solution soit sécurisée de bout en bout pour que, dès le début, nous puissions diffuser son projet sans risque. Les solutions de sécurité adoptées sont toujours à la pointe des standards du secteur.

La facilité de configuration d'une solution associée à des capteurs, actionneurs et modules de communication fait que la création d'une nouvelle activité pédagogique peut se faire directement sur un smartphone. Cette solution n'aura pas la prétention de répondre à des besoins trop complexes intégrant un grand nombre de variétés de capteurs ou encore des besoins nécessitant une intégration au système d'information de l'école.

Architecture

La solution Magnetics est basée sur de nombreuses briques techniques. Le choix de chacune de ses briques a été fait à la place de l'utilisateur afin qu'il puisse se concentrer sur les usages qu'il souhaite démontrer plus que sur des choix techniques multiples, complexes et surtout sans grande valeur dans les premières étapes du projet.

Les briques d'une solution IoT sont :

- Les capteurs
- Actionneurs
- Dispositifs principaux (carte programmable, passerelle, PC, smartphone,)
- Plateformes cloud
- Logiciels externes (applications d'utilisateur final ou tout autre composant logiciel).

En termes de communication, il existe au moins deux mondes différents. Dans le premier, le matériel utilise plusieurs protocoles pour échanger des données entre tous les composants. Pour communiquer directement entre une carte programmable et un capteur, nous utilisons le protocole Jacdac avec son câble à 3 fils. Lorsque nous devons connecter une carte programmable avec un ordinateur, nous transférons toute la trame Jacdac sur un canal USB CDC. La norme BLE sera utilisée pour que les capteurs, les actionneurs et les passerelles communiquent ensemble sans fil. Pour augmenter l'interopérabilité, nous définissons la couche JacdacBLEMesh pour transférer facilement les données entre le capteur, la carte programmable et les autres dispositifs principaux. Pour ces éléments, la consommation d'énergie est l'une des principales questions qui guideront la définition de la manière de communiquer. Une grande partie de cette question est déjà traitée de manière adéquate directement dans la norme BLE. Nous ne l'utiliserons donc qu'en intégrant nos contraintes de conception énoncées ci-dessus.

Le second univers est celui du cloud et il sera initialement principalement basé sur le protocole MQTT pour faire communiquer les différents composants logiciels. Toutes les capacités disponibles dans le matériel devront être rendues accessibles dans la partie cloud. Ainsi, toutes les actions, téléchargements de données ou paramétrages possibles dans l'univers du matériel devront impérativement être rendus accessibles dans l'univers du cloud à travers des topiques MQTT aussi proches que possible en termes de structure et de sémantique des services et des caractéristiques de l'équipement.

Certaines capacités supplémentaires peuvent être ajoutées du côté du cloud sans nécessiter l'ajout du côté du matériel. Par exemple, tous les calculs trop coûteux en termes de ressources pour être réalisés sur les capteurs ou la passerelle seront effectués directement par la plateforme. Pour rendre cela plus explicite d'un point de vue mathématique, l'application qui fait correspondre les caractéristiques de la cible au sujet MQTT doit impérativement être injective.

Pour des raisons d'efficacité énergétique, les communications doivent se faire autant que possible de manière asynchrone et en évitant le polling. Le matériel se réveille que périodiquement, c'est toujours lui qui restera maître du moment où il pourra répondre aux requêtes qui lui seront envoyées.

Le protocole JacdacBLEMesh

Pour permettre de détecter automatiquement les données transmises, le protocole a été défini pour être compatible avec le protocole filaire jcdac. Cette compatibilité, permet de créer des prototypes riches en capteurs comportant plusieurs cartes qui communiquent par les airs leurs données comme si elles étaient sur le même bus physique. Cette transparence référentielle permet d'ouvrir l'univers magnetics à l'ensemble des capteurs compatibles jcdac (<https://microsoft.github.io/jacdacs-docs/devices/>).

Spécifications des services

Les services sont spécifiés pour abstraire le dispositif matériel de l'implémentation logicielle. Les services sont constitués de registres, de commandes et d'événements, ainsi que d'informations précises sur la disposition des données pour chaque paquet.

Catalogue des appareils

Le catalogue des périphériques répertorie les périphériques enregistrés qui peuvent être détectés automatiquement sur le bus. Les informations du catalogue fournissent des informations sur le vendeur, les services pris en charge par un périphérique, le micrologiciel et les images. Ces informations sont également utilisées pour détecter automatiquement, télécharger et flasher les micrologiciels sur les appareils.

Protocole

Tous nos appareils communiquent à l'aide de paquets sur un "bus", où chaque appareil peut s'annoncer et présenter l'ensemble des services qu'il fournit. Un service fournit des registres, des événements et des commandes pour communiquer avec d'autres appareils. Les paquets sont envoyés en série entre les dispositifs physiques sur le bus et peuvent également être accessibles sur WebUSB/WebBLE, fournissant une connectivité à des outils et services basés sur le Web et fonctionnant dans le navigateur Web. La pile Jacdac de Microsoft sera au cœur de la communication dans le monde embarqué.

Jacdac est un nouveau protocole conçu pour faciliter la connexion de microcontrôleurs et de capteurs à faible coût. Jacdac n'est pas seulement conçu pour faciliter la connectivité physique, il s'agit d'une pile matérielle et logicielle complète qui relie le monde du microcontrôleur à faible coût au navigateur web et au-delà.

Le protocole principal de Microsoft est étendu pour être adapté au scénario sans fil. Ces variantes de Jacdac seront appelées JacdacBLE et JacdacBLEMesh.

Couches

Le protocole se compose de trois couches principales, décrites ci-dessous.

- Couche service : comment les appareils partagent les ressources matérielles ou logicielles entre eux ?

Les dispositifs Jacdac partagent leurs ressources avec d'autres dispositifs sur le bus par le biais de services. Les services fournissent des interfaces abstraites et standardisées qui peuvent être utilisées pour interagir avec des ressources matérielles physiques (par exemple un accéléromètre) ou des ressources purement virtuelles (par exemple l'état d'un jeu vidéo). Cette abstraction apporte un dynamisme "plug-and-play" au Jacdac, de sorte que des dispositifs avec un matériel différent, mais la même fonctionnalité globale, peuvent se remplacer les uns les autres sans avoir à recompiler les applications utilisateur. Par exemple, deux modèles différents d'accéléromètres peuvent se remplacer l'un l'autre car ils partagent la même interface logicielle.

Tout appareil qui héberge un service doit également exécuter le service de contrôle. Le service de contrôle est chargé d'annoncer tous les 500 millisecondes les services qu'un périphérique exécute. Comme tout autre service, les paquets émis par le service de contrôle sont normalisés. Il existe également un ensemble de commandes communes qui doivent être implémentées par les appareils Jacdac qui exploitent des services, y compris des fonctionnalités comme resetet time since boot.

La plupart des utilisateurs de Jacdac n'auront jamais besoin d'écrire un service : le principal cas d'utilisation est que les utilisateurs écrivent des applications qui interagissent avec les dispositifs et les services Jacdac. L'utilisation de Jacdac dans les applications de microcontrôleur est incroyablement facile et ne nécessite qu'une pile logicielle ayant une couche physique Jacdac compatible.

- Couche de transport : responsable de l'acheminement fiable des paquets vers et depuis les services et les applications

Chaque fois qu'une trame est reçue par la couche physique, la couche transport divise cette trame en paquets de données et les transmet au service ou à l'application utilisateur approprié. À ce stade, les services et les applications peuvent utiliser les paquets pour effectuer des actions basées sur de nouvelles données.

La présence ou l'absence d'un dispositif sur le bus est signalée par la présence ou l'absence de paquets publicitaires contenant l'identifiant du dispositif, qui doivent être envoyés toutes les 500 millisecondes. Si le client d'un dispositif Jacdac ne reçoit pas d'annonce du dispositif pendant plusieurs périodes d'annonce (1 à 2 secondes), il doit supposer que le dispositif a quitté le bus. Si le dispositif rejoint le bus, il peut présenter une liste de services différente. En fait, les dispositifs sont libres de modifier la liste des services qu'ils annoncent à tout moment.

Dans de nombreux cas, il est important que les données soient reçues par un dispositif spécifique. Jacdac supporte ce mécanisme en utilisant des accusés de réception et des tuyaux. Les accusés de réception sont incroyablement simples et nécessitent que les appareils récepteurs reconnaissent qu'une trame a été reçue avec succès. Les tuyaux, quant à eux, établissent une connexion point à point entre les appareils, et chaque paquet est acquitté et reçu en utilisant une approche de fenêtre glissante. Seuls les appareils les plus performants sont censés prendre en charge les tuyaux. JacdacBLE et JacdacBLEMesh utilisent les mécanismes de la norme BLE pour prendre en charge la publicité, l'accusé de réception et les tuyaux. Le paquet transmis par un appareil reste inchangé entre les différents types de couches physiques.

- Couche physique : transmet/reçoit des paquets vers/depuis d'autres dispositifs Jacdac

Il peut y avoir plusieurs implémentations de la couche physique responsables de l'envoi de trames sur un support de communication câblé ou sans fil. Les trames contiennent un ou plusieurs paquets, et les paquets contiennent une commande ou un rapport de service. Les trames contiennent également des métadonnées d'adressage que la couche transport utilise pour acheminer correctement les paquets.

Trames, paquets et indices de service

Une trame Jacdac contient une liste de paquets Jacdac (d'une longueur d'au moins un), qui partagent tous le même identifiant de dispositif et les mêmes indicateurs (voir la description logique d'un paquet Jacdac ci-dessous). La trame contient également un code de redondance cyclique (CRC). Si l'identifiant du dispositif de la trame correspond à l'identifiant du dispositif, alors le dispositif envoie un accusé de réception contenant le CRC de la trame.

Paquet

```
struct {
    uint8_t flags;
    uint64_t device_identifier;
    uint8_t service_size;// size of the data payload
    uint8_t service_index;// the index into the device's advertised list of
services
    uint16_t service_command;
    uint8_t data[0];
}
```

Un dispositif maintient généralement un état spécifique au dispositif pour chacun des services qu'il prend en charge. Cet état devrait être conservé dans un tableau, où chaque entrée du tableau contient la classe de service et (un pointeur vers) l'état spécifique au périphérique pour ce service. Lorsqu'un dispositif annonce ses classes de service, elles sont présentées dans le même ordre que dans la liste. Cela permet aux clients du périphérique de se référer à l'un de ses services par l'index de la liste (service_index, une quantité de 8 bits) au lieu de la classe de service (une quantité de 32 bits). L'index zéro (numéro de service) est réservé au service de contrôle.

Flags

Un paquet ne contient qu'un seul identifiant de périphérique (plutôt que les identifiants de source et de destination, comme dans IP). Si le bit le plus bas de flags est activé, le paquet est un paquet de commande et device_identifier est le périphérique de destination qui reçoit le paquet ; sinon, le paquet est un paquet de rapport et device_identifier est le périphérique source qui diffuse les informations sur le bus.

Frame

Les dispositifs Jacdac communiquent sur le "bus" via des trames. Une trame contient un identifiant de périphérique et un ou plusieurs paquets et a la structure suivante : Sur le fil, la trame doit être transmise au format little-endian (c'est-à-dire l'octet de poids faible de frame_crcen premier). La taille maximale totale de la trame est de 252 octets, choisie pour garder la taille totale du paquet sous 255 (la limite DMA sur certains matériels) et alignée sur 4. Le tableau suivant définit la signification et la taille des champs dans la structure ci-dessus.

```
struct{
    uint16_t frame_crc;
    uint8_t frame_size;
    uint8_t frame_flags;
    uint64_t device_identifier;
    uint8_t data[240]; // maximum
}
```

Frame flags

Le champ frame_flags est utilisé pour indiquer :

0	COMMAND	Si elle est définie, la trame contient des paquets de commande, si elle n'est pas définie, la trame contient des paquets de rapport.
1	ACK_REQUESTED	Si elle est définie, le récepteur doit répondre à l'expéditeur avec une trame ACK, si elle n'est pas définie, aucune réponse n'est requise.
2	DEVICE_ID_ALT_MEANING	
3-7	RESERVED	Réserve pour une utilisation future.

Les paquets sont placés dos à dos à l'intérieur du champ de données de la trame et doivent être remplis de manière à commencer à une frontière de 4 octets (c'est-à-dire que la taille du service est divisible par 4). Logiquement, les paquets placés dans la même trame partagent les mêmes champs de trame et donc le même identifiant de périphérique. Cela signifie qu'un seul dispositif peut être adressé dans une même trame.

Couches d'abstraction matérielle pour l'interopérabilité entre les plateformes

La carte utilisée pour prototyper et qui nous a été mis à disposition en grande quantité par l'entreprise STMicroelectronics, est le kit de développement B-L475E-IOT01A Discovery kit for IoT node. Cette carte dispose de nombreux capteurs (centrale inertielle 9 axes, humidité, pression, température, distance,...) ainsi que des capacités de communication multiples (833MHz, Wifi, BLE).

Pour pouvoir être utilisé dans les plateforme de programmation, il fallait disposer de pilotes pour chacun des périphériques de la carte. Le développement d'une couche d'abstraction matérielle permet d'éviter de développer trois fois cet ensemble de pilote. Les besoins des projets a conduit au développement d'un tel ensemble de composants logiciel et disponible à l'adresse suivante : <https://github.com/letssteam/codal-stm32>

Le runtime codal fournit un environnement facile à utiliser pour programmer la carte en langage C/C++. Il contient des pilotes de périphériques pour toutes les fonctionnalités matérielles, ainsi qu'une suite de mécanismes d'exécution pour rendre la programmation plus simple et plus flexible. Ceux-ci vont du contrôle de l'affichage matriciel à LED à la communication radio peer-to-peer et aux services sécurisés Bluetooth Low Energy.

En plus de prendre en charge le développement en C/C++, le runtime est également conçu spécifiquement pour prendre en charge les langages de niveau supérieur fournis par nos partenaires qui ciblent l'informatique physique et l'enseignement de l'informatique.

Codal-core doit être implémenté par un développeur tiers pour prendre en charge la nouvelle cible matérielle. Le référentiel codal-stm32 contient l'implémentation supportant l'intégralité des microcontrôleurs de la famille des STM32.

Ce portage est basé sur :

STM32Cube comprenant :

- La couche d'abstraction matérielle HAL, permettant la portabilité entre différents appareils STM32 via des appels d'API standardisés
- Les API Low-Layer (LL), un ensemble d'API léger, optimisé et orienté expert conçu pour les performances et l'efficacité de l'exécution

Définition d'appareil CMSIS pour STM32 CMSIS :

Cortex Microcontroller Software Interface Standard (CMSIS) est une couche d'abstraction matérielle indépendante du fournisseur pour la série de processeurs Cortex®-M et définit des interfaces d'outils génériques. Il a été emballé sous forme de module pour Arduino IDE : <https://github.com/stm32duino/ArduinoModule-CMSIS>

Chaîne d'outils GNU Arm Embedded

Compilateur Arm Embedded GCC, bibliothèques et autres outils GNU nécessaires au développement de logiciels bare-metal sur des appareils basés sur Arm Cortex-M. Les packages sont fournis grâce au xPack GNU Arm Embedded GCC : <https://github.com/xpack-dev-tools/arm-none-eabi-gcc-xpack>

Le moteur d'exécution est constitué de plusieurs bibliothèques. La bibliothèque commune à toutes les cibles Codal est connue sous le nom de [codal-core] (<https://github.com/lancaster-university/codal-core>). Cette bibliothèque contient les pilotes communs, les abstractions pour les pilotes communs (modèles de pilotes), l'ordonnanceur et les mécanismes d'événements, essentiels à l'expérience Codal.

Les cibles sont des bibliothèques avec une addition, un fichier appelé target.json. Le système de construction Codal utilise les informations de ce fichier pour sélectionner et configurer les chaînes d'outils, et télécharger les dépendances. target.json contient également les informations nécessaires pour configurer les bibliothèques, comme mbed, ainsi que pour fournir des définitions spécifiques aux périphériques pour codal-core. Le target.json peut également configurer une tâche de post-traitement, exécutée depuis la racine de Codal. Pour cet aspect, notre travail est divisé en plusieurs sous-projets publiés sur plusieurs dépôts github :

codal-core (<https://github.com/letssteam/codal-core>)

La bibliothèque centrale de codal. Au cours du projet, nous apportons des contributions à ce dépôt. Principalement pour corriger des problèmes, mais aussi pour ajouter une certaine abstraction qui pourrait être utilisée par toutes les autres cibles.

codal-stm32 (<https://github.com/letssteam/codal-stm32>)

La bibliothèque commune pour toutes les cibles STM32. Dans ce dépôt, nous avons créé le pilote commun pour tous les microcontrôleurs STM32. Cette bibliothèque est particulièrement importante car elle est le lien entre le HAL STM32 de très bas niveau et Codal. Nous avons défini un mécanisme appelé variant qui permet de faire ce lien. Un variant est une configuration qui définit toutes les spécifications d'un microcontrôleur.

codal-stm32-DISCO_L475VG_IOT (https://github.com/letssteam/codal-stm32-DISCO_L475VG_IOT)

Ce référentiel est la cible codale pour le B-L475E-IOT01A Kit de découverte pour nœud IoT. Il fait le lien entre le mécanisme de variante et l'abstraction codale. Les pilotes spécifiques à la carte sont contenus dans ce référentiel.

codal-stm32-PNUCLEO_WB55RG (https://github.com/letssteam/codal-stm32-PNUCLEO_WB55RG)

Ce dépôt est la cible codale pour la carte P-NUCLEO-WB55. Nous utilisons cette cible lors de la conception de la carte STEAM32.

codal-stm32-NUCLEO_F4x1RE (https://github.com/letssteam/codal-stm32-NUCLEO_F4x1RE)

Ce référentiel est la cible codale pour la carte NUCLEO_F4x1RE. Nous avons développé cette cible spécifiquement pour la collaboration avec Perlatechnica sur roboopoly.

codal-stm32-STEAM32_WB55RG (https://github.com/letssteam/codal-stm32-STEAM32_WB55RG)

Ce dépôt est la cible codal pour la carte STEAM32. Depuis le début des projets Let's STEAM, nous préparons toujours toutes nos contributions pour être compatibles avec cette carte. Même si elle n'est pas disponible commercialement maintenant, la carte STEAM32 sera un vrai changement de jeu qui aidera grandement la diffusion et la réPLICATION des résultats faits pendant le projet Let's STEAM.

codal-letssteam (<https://github.com/letssteam/codal-letssteam>)

Ce dépôt est l'outil destiné à faciliter la vie de l'équipe de développement de Let's STEAM. Il contient quelques scripts pour soutenir le cycle de vie d'un projet codal avec de nombreuses cibles. Au début, le dépôt était principalement utilisé pour le processus d'intégration continue mais maintenant il est aussi devenu un moyen pratique de contribuer au projet Let's STEAM.

DapLink

Dans le cadre du projet magnetics, il était primordial de proposer aux utilisateurs une expérience similaire quels que soient les outils. Cependant, le STM32L475 Discovery nécessite un téléchargement manuel du programme, ce qui est contraignant puisqu'il est nécessaire de le pré-programmer la carte en fonction de l'outil utilisé (MakeCode, MicroPython ou Scratch). Enfin, il était crucial de pouvoir échanger des données (texte, valeurs numériques, ...) avec la carte lors de l'exécution d'un programme, ce qui ne pouvait se faire directement.

C'est pourquoi nous avons choisi de mettre en place un micrologiciel basé sur DAPLink. DAPLink est projet open-source porté par ARM, dans le cadre du développement de sa plateforme MBed. Cet outil vise à permettre la programmation et le débogage (test, recherche, et correction de 'bug' in situ), d'une carte électronique.

A l'aide de DapLink et de la technologie "WebUSB" (permettant d'accéder au périphérique USB via un navigateur web), il est possible maintenant de programmer et d'échanger des données (communication série) sans disposer d'outils tiers. De plus, cela permet une interopérabilité entre les différentes plateformes (MakeCode, MicroPython, Scratch).

Le micrologiciel vient en lieu et place du STLink présent par défaut sur toute les cartes ST. Les fonctionnalités de base ont été implémentées, mais pas le WebUSB. De plus, nous avons découvert quelques bogues critiques que nous avons dû corriger. Les corrections de bogues et le portage de la carte sur DapLink ont été partagés avec la communauté pour que le micrologiciel puisse être utilisé par d'autres acteurs sur n'importe quelle carte STM32 : <https://github.com/letssteam/DAPLink>.

Maillage numérique

Makecode

MakeCode est le nom commun de plusieurs éditeurs de programmation basés sur des blocs. Tous ces éditeurs sont basés sur un cadre commun appelé PXT (Programming Experience Toolkit).

PXT est un cadre permettant de créer des expériences de programmation spécifiques pour les débutants, en particulier pour l'enseignement de l'informatique. Le langage de programmation sous-jacent de PXT est un sous-ensemble de TypeScript (sans les fonctionnalités dynamiques de JavaScript).

Les principales caractéristiques de PXT sont les suivantes

- un éditeur de code basé sur Blockly et un convertisseur au format texte
- un éditeur de code Monaco qui alimente [vs Code] (<https://github.com/microsoft/vscode>), les caractéristiques de l'éditeur sont énumérées ici : <https://code.visualstudio.com/docs/editor/editingevolved>.
- un support d'extensibilité pour définir de nouveaux blocs en TypeScript
- un émetteur de code machine ARM Thumb
- un gestionnaire de paquets en ligne de commande

Pour cet aspect, notre travail est divisé en plusieurs sous-projets publiés sur plusieurs dépôts github :

- pxt (<https://github.com/letssteam/pxt>) : ce dépôt contient la version spécifique de pxt utilisée par le projet Let's STEAM. Nous l'utilisons principalement lorsque nous avons rencontré des problèmes que nous avons corrigés en faisant des PR.
- pxt-common-package (<https://github.com/letssteam/pxt-common-packages>) : Ce dépôt est un ensemble de paquets utilisés dans divers éditeurs MakeCode. Nous l'utilisons principalement lorsque nous avons rencontré des problèmes que nous avons corrigés en faisant des PR.
- pxt-magnetics (<https://github.com/letssteam/pxt-magnetics>) : Ce dépôt contient l'extension MakeCode pour utiliser Magnetics dans un éditeur tier.
- pxt-lets-steam (<https://github.com/letssteam/pxt-lets-steam>) : Ce dépôt contient tout le code de l'éditeur MakeCode Let's STEAM. Le simulateur pour les cartes STM32 se trouve ici. On peut y trouver le code de collage qui fait le lien entre les abstractions codales et le langage de blocs.

-
- pxt-lets-steam-static (<https://github.com/letssteam/pxt-lets-steam-static>) : Ce dépôt contient tous les fichiers HTML de l'éditeur Let's STEAM actuel disponible sur <https://makecode.lets-steam.eu/>.
 - pxt-lets-steam-backend (<https://github.com/letssteam/pxt-lets-steam-backend>) : Ce dépôt contient le backend de pxt qui gère le calcul côté serveur utilisé par l'éditeur makecode.
 - makecode-lets-steam (<https://github.com/letssteam/makecode-lets-steam>) : Ce dépôt est le dépôt principal de l'éditeur MakeCode Let's STEAM. Il fait le lien avec tous les dépôts précédents et fournit des outils pour exécuter facilement l'éditeur.
 - makecode-toolchain-docker(<https://github.com/letssteam/makecode-toolchain-docker>) : ce dépôt contient la définition des chaînes d'outils utilisées par makecode pour compiler un programme pour la carte électronique. Ceci est particulièrement important pour être sûr que toutes les instances de notre éditeur utilisent exactement la même toolchain.
 - makecode-lets-steam-container(<https://github.com/letssteam/makecode-lets-steam-container>) : Ce dépôt contient une définition de conteneur pour développer et contribuer facilement sur l'éditeur MakeCode de Let's STEAM.

Micropython

MicroPython est un projet open-source, visant à porter le langage Python sur des plateformes embarquées (cartes électroniques, plus ou moins complexes). Les principaux avantages de cet outil sont :

- L'interrogeabilité du code Python, sur toutes les cartes électroniques supportées par MicroPython.
- La boucle "Read-Eval-Print Loop" (REPL), permet à l'utilisateur d'écrire des lignes de code individuelles et de les exécuter immédiatement, et d'observer le résultat. Ceci est utile aux débutants pour apprendre le langage.
- Les "modules" sont des fonctionnalités pré-compilées et testées qui peuvent être partagées par la communauté pour ajouter le support de différents capteurs, actionneurs, écrans, etc....

Le kit STM32L475 Discovery et la STEAM32 ont été ajoutés sans trop de complexité à Micropython car le support des cartes STM32 est natif dans le projet. En effet, la première carte pour laquelle ce projet a été développé, la pyboard, utilisait un microcontrôleur STM32 et les développeurs d'origine avaient mis en place tous les éléments de base pour cette famille de composant.

Il a été nécessaire de développer et d'intégrer les pilotes de tous les capteurs (internes ou externes) utilisés dans les fiches d'activités dans notre version de MicroPython.

Pour mettre en oeuvre facilement notre version de micropython, un l'éditeur web simpliste a été développé. Il permet à l'utilisateur d'écrire du Python, dans son navigateur, et de fournir une expérience de type MakeCode pour télécharger et échanger des informations avec le kit STM32L475 Discovery et la STEAM32.

Pour cet aspect, notre travail est divisé en plusieurs sous-projets publiés sur plusieurs dépôts github :

- [micropython-magnetics](https://github.com/letssteam/micropython-magnetics) (<https://github.com/letssteam/micropython-magnetics>) : Ce dépôt contient l'intégration de la bibliothèque magnetics au cœur de micropython
- [micropython-DISCO_L475VG_IOT-lib](https://github.com/letssteam/micropython-DISCO_L475VG_IOT-lib) (https://github.com/letssteam/micropython-DISCO_L475VG_IOT-lib) : Ce dépôt rassemble l'ensemble des pilotes pour les capteurs de la carte STM32L475 Discovery. Il est utilisé pour créer le firmware qui sera utilisé comme base pour travailler avec micropython sur les carte STM32.
- [micropython-editor-experimental](https://github.com/letssteam/micropython-editor-experimental) (<https://github.com/letssteam/micropython-editor-experimental>) : Ce dépôt contient l'éditeur web permettant d'utiliser magnetics sur les cartes STM32.
- [micropython-microbit-v2](https://github.com/letssteam/micropython-microbit-v2) (<https://github.com/letssteam/micropython-microbit-v2>) : Ce dépôt contient l'implémentation de micropython pour la microbit V2. Cette version basée sur codal, est nécessaire pour utiliser les fonctionnalités BLE de cette carte.

Scratch

Enfin, le travail réalisé sur Scratch a porté tout d'abord sur une étude d'usabilité du logiciel dans le cadre de l'approche par enquête prônée au sein de Let's STEAM. L'éditeur Scratch étant principalement centré sur le pilotage des cartes et non leur programmation, l'utilisation de cet outil est assez limitée. De plus, le modèle de développement de Scratch ne permet pas aux utilisateurs externes au Lifelong Kindergarten du MIT de proposer des extensions à l'éditeur officiel. La plupart du temps, pour proposer un contenu particulier, centré sur un usage particulier, il faut créer et héberger un nouvel éditeur spécifique. Ce nouvel éditeur, non lié à l'éditeur officiel a tendance à fragmenter le marché et perd grandement les utilisateurs enseignants et apprenants. Cependant, afin d'explorer les synergies avec une solution logicielle utilisée par les enseignants, particulièrement en début de cycle secondaire, un travail d'identification de synergies a été mené.

Cela a permis d'identifier un éditeur Scratch auquel nous pouvions ajouter nos fonctionnalité, porté par la société Vittascience, avec qui plusieurs réunions de travail ont été organisées pour intégrer des fonctionnalités magnetics à cet éditeur. Comme cet éditeur utilise Micropython pour communiquer avec la carte, l'intégration logicielle n'a demandé que peu de développements additionnels. En effet, de notre côté, il nous a suffi de réécrire le script Micropython existant pour micro:bit pour supporter nos particularités et nos capteurs. Nous sommes actuellement en attente du déploiement de la nouvelle version de l'éditeur Vittascience pour la STEAM32 pour rendre disponible nos résultats. Travaillant avec l'entreprise responsable de cet éditeur sur le projet SteamCity, nous avons intégré ce déploiement dans ce nouveau projet pour faire en sorte de les coûts engendrés pour ce partenariat puisse être couvert en dehors de l'enveloppe de magnetics qui ne l'avait pas prévu initialement.

ouverture

L'Open Source Hardware Association

L'Open Source Hardware Association (OSHWA) est une organisation à but non lucratif qui défend le matériel open source, à travers la création d'un hub d'activités liées au matériel open source. Elle a été créée en tant qu'organisation en juin 2012 par l'ingénieur Alicia Gibb, qui avait travaillé sur l'Open Hardware Summit pendant ses études supérieures.

L'OSHWA a pour objectif de favoriser les connaissances technologiques et d'encourager une recherche accessible, collaborative et respectueuse de la liberté des utilisateurs. Les principales activités de l'OSHWA comprennent l'organisation de l'Open Hardware Summit annuel et le maintien de la certification Open Source Hardware, qui permet à la communauté d'identifier et de représenter rapidement le matériel conforme à la définition communautaire du matériel open source.

Qu'est-ce qui est considéré comme du "matériel open source" ?

Le matériel open source est un terme désignant des objets physiques - machines, dispositifs ou autres choses physiques - dont la conception a été rendue publique de manière que tout le monde puisse les fabriquer, les modifier, les distribuer et les utiliser. Par conséquent, le matériel à source ouverte consiste à permettre à d'autres de construire et d'améliorer le matériel existant.

L'Open Source Hardware Association

Mettre ces objets à la disposition des autres nécessite deux étapes : 1) mettre à la disposition du public les fichiers sources, les nomenclatures et d'autres informations sur le matériel, afin que les utilisateurs comprennent le fonctionnement du matériel, et 2) accorder une licence pour le matériel de manière à permettre à d'autres d'utiliser et de développer ces informations. Idéalement, le matériel à source ouverte utilise des composants et des matériaux facilement disponibles, des processus standards, une infrastructure ouverte, un contenu sans restriction et des outils de conception à source ouverte afin de maximiser la capacité des individus à fabriquer et à utiliser du matériel. Dans le domaine industriel, le matériel à source ouverte donne aux gens la liberté de contrôler leur technologie tout en partageant les connaissances et en encourageant le commerce par l'échange ouvert de conceptions.

La certification OSHWA

La certification OSHWA offre aux producteurs un moyen simple et direct d'indiquer que leurs produits sont conformes à une norme uniforme et bien définie en matière de matériel à code source ouvert. La certification OSHWA pour le matériel open source donne aux utilisateurs la certitude que la définition du "matériel open source" utilisée par un projet spécifique correspond à la définition communautaire de l'open source. Découvrez la liste complète du matériel open source certifié par l'association OSHWA ici : <https://certification.oshwa.org/list.html>



Contactez-nous et découvrez l'ensemble de nos ressources



Posez-nous des questions

Contactez-nous : contact@labaixbidouille.com

Partagez vos commentaires et corrections

Ce guide a été réalisé avec la meilleure qualité possible et une réelle volonté de participer à l'émergence de contenus stimulants dans le domaine de la programmation. Cependant, nous ne sommes que des humains ! Si vous découvrez des erreurs ou des corrections à apporter, n'hésitez pas à nous contacter ! Nous ferons en sorte que vous soyez crédité·e pour votre aide !

Coopérer avec nous au sein de nouveaux projets

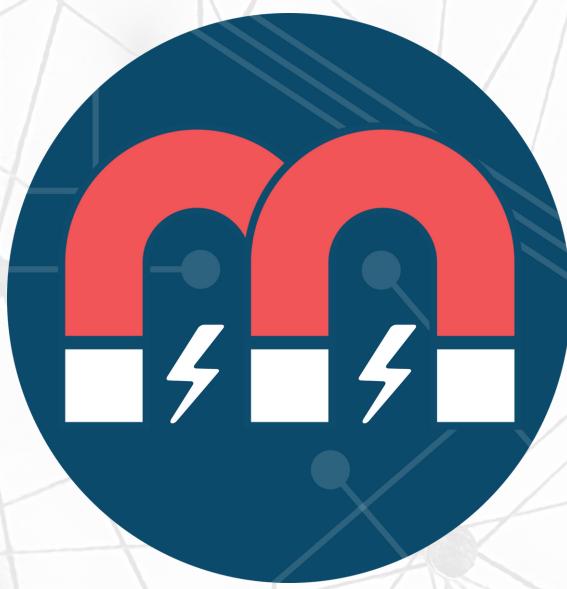
Le L.A.B lance régulièrement de nouveaux projet et est ouvert à de nouvelles coopérations, que ce soit avec des écoles mais aussi avec des les acteurs du monde de la programmation. N'hésitez pas à nous contacter pour construire ces initiatives ensemble !

Explorer nos ressources

- magnetics : <https://www.magnetics.edu-up.fr/>
- Let's STEAM : <https://lets-steam.eu/>
- TheDexterLab :
<http://www.thedexterlab.eu/resources>
- Notre wiki Les Emulsionneurs :
<https://lesemulsionneurs-wiki.notion.site>

Découvrir le dispositif edu-up

<https://eduscol.education.fr/1603/le-dispositif-edu>



magnetics

[HTTPS://WWW.MAGNETICS.EDU-UP.FR/](https://www.magnetics.edu-up.fr/)