Créer un jeu vidéo et des commandes faites « maison » pour le contrôler

E:\GD-PARENTS\JEU_VIDEO_ELECTRICITE\FICHE_ANIMATION

Cet atelier permet d'apprendre à :

- Programmer un jeu vidéo

- Construire un dispositif physique pour contrôler un élément graphique (Sprite) dans ce jeu.

Vous programmerez par blocs dans Scratch afin de concevoir le « gameplay » d'un jeu de voiture. Ensuite vous découvrirez comment construire un dispositif de contrôle de jeu composé d'un volant et de pédales à partir de fils électriques et de morceaux de bois.

Cet atelier s'adresse principalement aux néophytes et débutants. Ce document présente également des extensions de l'atelier qui peuvent être réalisées à domicile avec des enfants et petits-enfants afin de mobiliser des compétences travaillées pendant l'atelier et de progresser jusqu'à un niveau intermédiaire.

Cette séquence a été conçue et produite par :

Ce document est diffusé sous licence Creative Commons

Vous êtes autorisé·e à partager (copier et redistribuer) le matériel à condition de respecter les termes énoncés ci-dessous.

Attribution — Vous devez créditer l'œuvre de manière appropriée (en indiquant le nom du projet et l'organisme de financement), fournir un lien vers la licence et indiquer si des modifications ont été apportées. Vous pouvez le faire de toute manière raisonnable, mais sans suggérer que l'auteur ou le licencié vous soutient ou approuve l'usage que vous faites de l'œuvre.

Pas d'utilisation commerciale — Vous n'êtes pas autorisé·e à faire un usage commercial de ce matériel, en tout ou partie.

Pas de modification — Si vous modifiez, transformez ou adaptez ce matériel, vous n'êtes pas autorisé·e à diffuser le matériel modifié.

Table des matières

1. Durée de l'activité 4
2. Matériel nécessaire pour réaliser cette activité 4
3. Recommandations destinées à l'animateur·rice de l'atelier 4
4. Déroulement de l'atelier 5
4.1. Découvrir Scratch 5
4.2. Programmer les prémices de notre jeu. 9
4.3. Construire une pédale de gaz pour contrôler la voiture dans le jeu 9
4.4. Utiliser la pédale comme un émulateur de clavier 10
5. Extensions 13
5.1. Comment simuler un clavier avec une carte éléctronique micro:bit et un circuit électrique pour interagir avec un ordinateur ? 13
Introduction 13
Objectif de cette extension 13
Description du programme 13
5.2. Ajouter une variable dans le programme pour collecter des infos sur les déplacements de la voiture. 17
Introduction 17
Objectif de l'extension 17
Exercices pour comprendre et intégrer la notion de variable 17
Solutions des exercices 18
5.3. Utiliser l'accéléromètre de la carte électronique pour diriger la voiture 19
Rappel 19
Objectif de l'extension 19
Description du programme 19
5.4. Programmer la version « complète » du jeu. 21
Rappel 21
Objectif de l'extension 21
Description du programme 21

Version provisoire

1. Durée de l'activité

La durée de l'atelier est estimée à 2,5 heures. Chaque extension prend de 30 min à 1 heure. Celles-ci peuvent être réalisées en autonomie, à domicile.

2. Matériel nécessaire pour réaliser cette activité

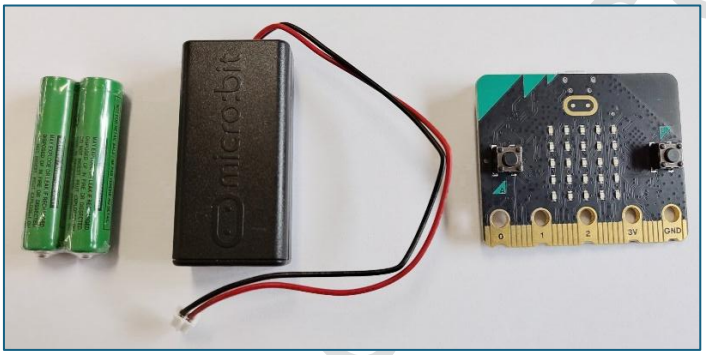
Par participant :

> 1 ordinateur (ou tablette) avec l'application (gratuite) Scratch.

> 1 plaque en bois d'environ 25 cm de côté et de 3-4 mm d'épaisseur.

> 0,5 mètres de fil électrique

> 1 carte micro:bit et sa batterie



> 1 charnière (récupérée sur une porte d'armoire ou imprimée en 3D)

> 1 ressort de 1-2 cm de haut.

NB : Le matériel utilisé pour fabriquer la pédale est illustré en détail dans les annexes 4.

3. Recommandations destinées à l'animateur·rice de l'atelier

 # Mettre l'accent sur les concepts informatiques (programmation séquentielle et événementielle, boucle, condition, variable) **et scientifiques** (circuit électrique ouvert/fermé, matériau conducteur/isolant, circuit en série et parallèle) lors de la conduite de l'atelier. Les participants saisiront mieux comment transférer leurs apprentissages à d'autres activités, notamment avec leurs enfants et petits-enfants.

 # Proposer des explications personnalisées en variant les approches. Il est possible d'expliquer les concepts de programmation en utilisant des activités déconnectées, en traçant des logigrammes, en proposant des exercices dirigés sur des applications. Plusieurs exemples concrets sont explicités dans les fiches pédagogiques des séquences de cours ScienceNum. Ces fiches sont disponibles gratuitement à l'adresse : https://hypothese.be/projet/science-num/.

Version Provisoire

 # Insister sur les invariants du numérique. Il est intéressant de montrer que les fonctions courantes sont représentées par des icônes très similaires dans la plupart des applications. À titre d'exemple, téléverser (upload) est généralement représenté par une ligne horizontale surmontée d'une flèche pointant vers le haut.

 # Expliquer les anglicismes et la raison de leur utilisation. Dans cette fiche, certains termes anglais sont utilisés sans être traduits en français. Il est utile d'expliquer que ces termes sont aussi utilisés par les développeurs et les joueurs francophones. Cela s'explique par le fait qu'il n'existe pas toujours de traduction adéquate et que l'utilisation du terme anglais fournit plus de résultats lors de recherches sur le Web.

4. Déroulement de l'atelier

4.1. Découvrir Scratch

\\ Niveau : néophyte

\\ Concepts & notions : interface, *sprites* et programmation par blocs

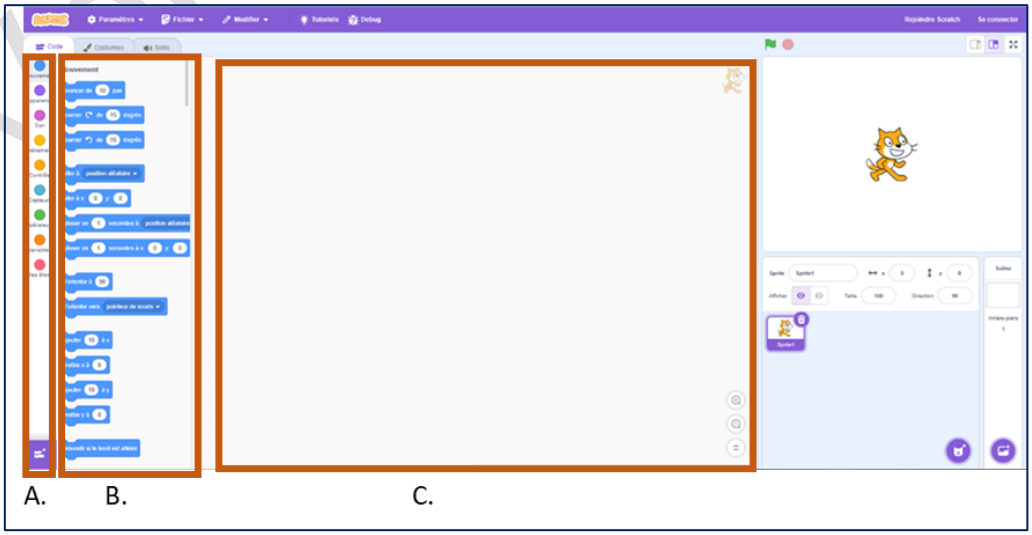
\\ Modalité : L'animateur montre puis les participants reproduisent par deux.

- L'animateur présente l'application Scratch :

« Scratch est un langage de programmation graphique à but éducatif. Il permet de créer des histoires interactives, des jeux et des animations en assemblant des blocs de code graphiques (contrairement à la programmation textuelle). Scratch est convivial et gratuit. Il favorise la pensée créative, le raisonnement systématique et la collaboration. »

- Il poursuit en décrivant les différentes parties de l'interface.

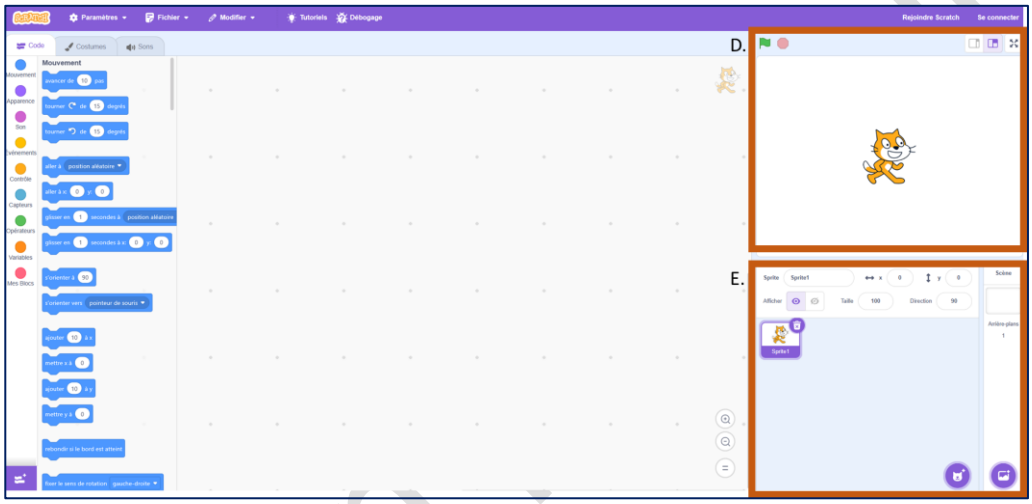
« A gauche, on trouve la zone dédiée à la programmation par bloc. Cette zone peut être divisée en trois sections. La section A reprend les bibliothèques de blocs. La section B, les blocs appartenant à la bibliothèque sélectionnée. La section C permet de construire le code du programme en assemblant des blocs. »



Version provisoire

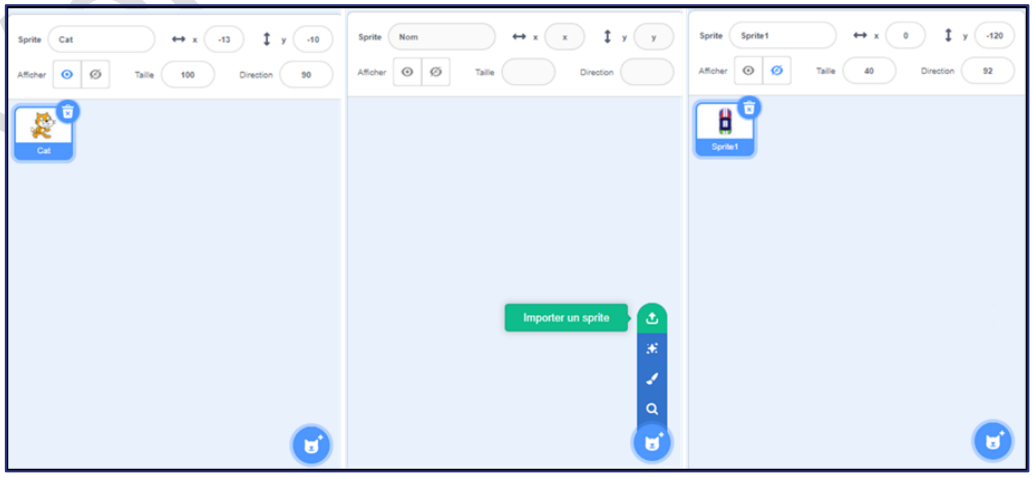
- L'animateur présente quelques blocs et les assemble dans la zone de programmation (section C). Par exemple, les blocs « Quand le drapeau vert est sélectionné » et le bloc « Avancé de X pas ».

- L'animateur enchaine en ajoutant que la partie en haut à droite (section D) de l'interface est une zone de prévisualisation. Elle permet de voir ce qui se passera dans le jeu que l'on programme.



- Il termine par la partie située en bas à droite (section E).

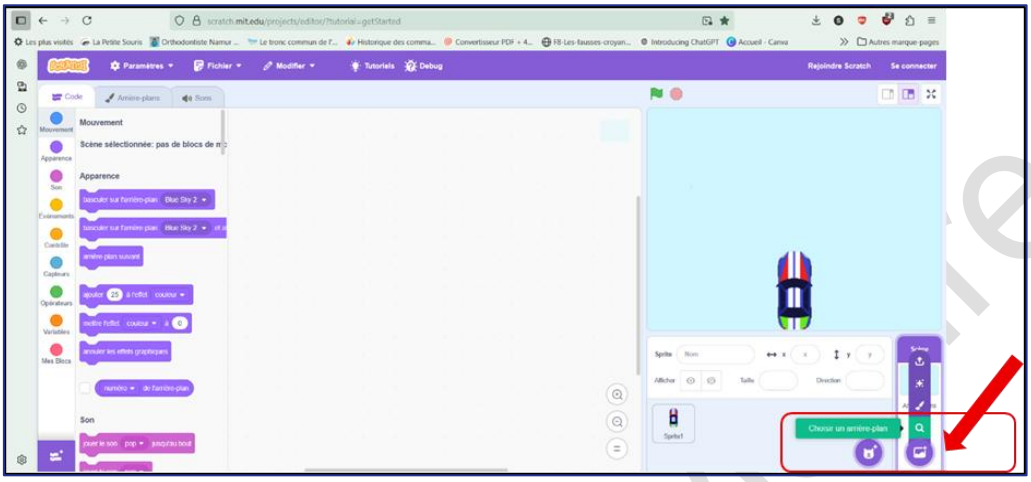
« Cette partie permet d'ajouter des éléments dans notre jeu. On parle de *sprites.* Je peux, par exemple, remplacer le sprite de chat par un *sprite* de voiture (Annexe 1) que j'ai sur mon ordinateur. Pour importer ce fichier, je sélectionne la barre horizontale surmontée d'une flèche vers le haut. »



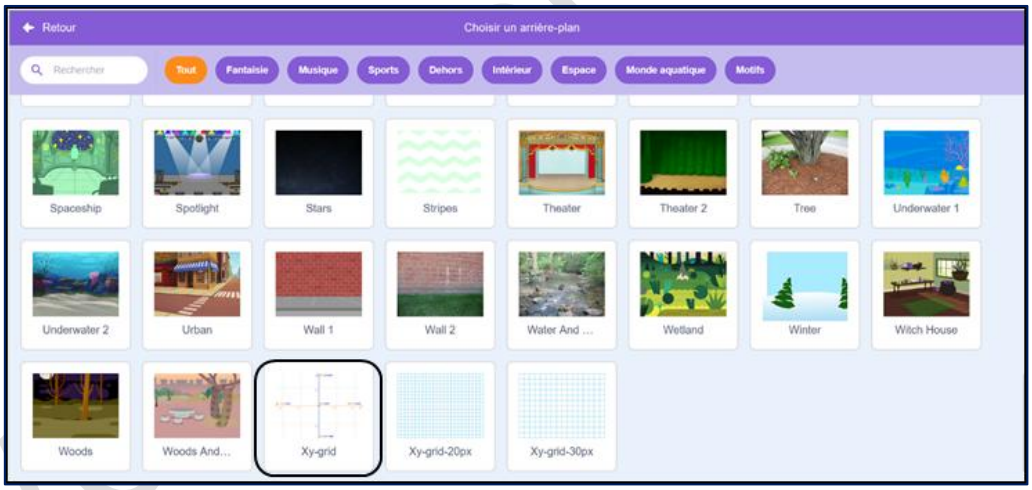
Version finale

- Il poursuit :

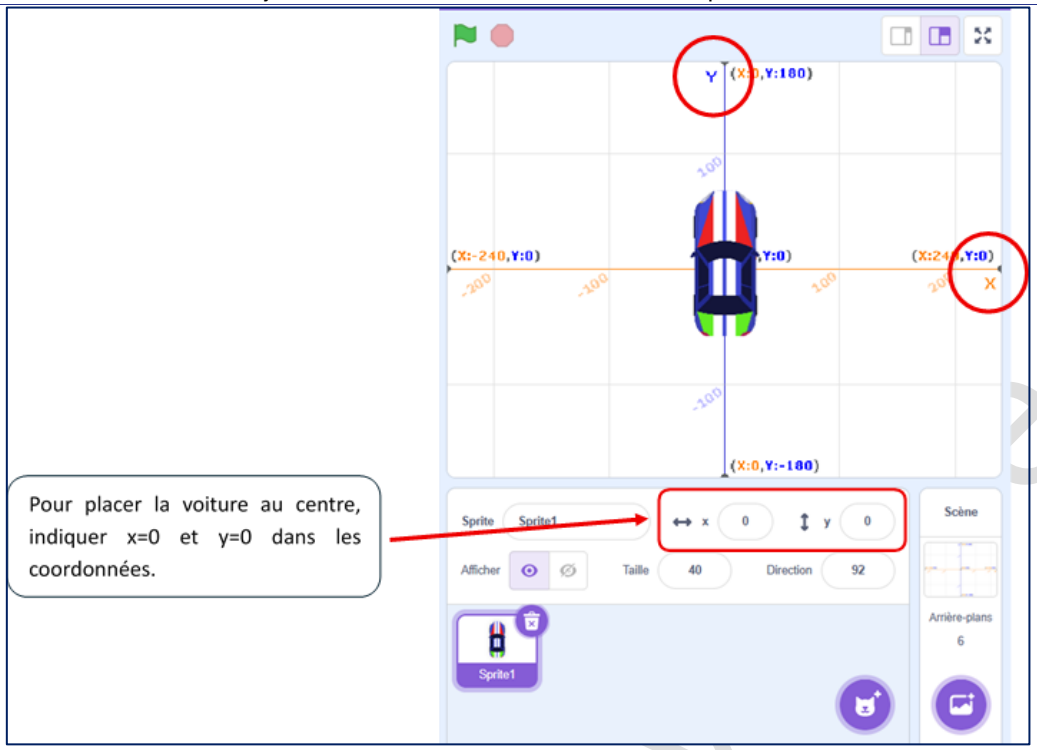
« Je peux également ajouter un arrière-plan issu de la bibliothèque d'arrière-plans. … Si je souhaite changer l'arrière-plan et placer un quadrillage pour mieux comprendre les déplacements de la voiture, je sélectionne la loupe dans le menu arrière-plan. …



… Je recherche et sélectionne l'arrière-plan **Xy – grid.** Cet arrière-plan constitue un repère qui contient deux axes : X et Y qui permettront de situer la voiture dans le plan. …



… Ce repère sera très utile lorsque nous voudront déplacer la voiture. L'axe X va permettre de déplacer la voiture de façon horizontale et l'axe Y de façon verticale. »



- L'animateur propose aux participants de reproduire ce qu'il a montré sur les ordinateurs

Version provisoire

4.2. Programmer les prémices de notre jeu.

\\ Niveau : débutant

\\ Concepts & notions : programmation séquentielle et événementielle, boucle et condition.

\\ Modalité : les participants travaillent en semi-autonomie en suivant les instructions du support (Annexe 2).

- L'animateur propose aux participants d'utiliser l'annexe 2 pour programmer les fonctions basiques du jeu, notamment le déplacement de la voiture. Le programme à obtenir correspond au fichier de l'annexe 3.
- L'animateur est disponible pour aider les participants qui avancent à leur rythme dans l'annexe 2.
- L'animateur prend quelques minutes pour formaliser les apprentissages en programmation.

4.3. Construire une pédale de gaz pour contrôler la voiture dans le jeu

\\ Niveau : néophyte

\\ Concepts & notions : circuit électrique d'un bouton poussoir, circuit ouvert/fermé.

\\ Modalité : les participants travaillent en semi-autonomie en suivant les instructions de l'annexe 4.

- L'animateur explique que la troisième étape est de construire un dispositif physique pour contrôler le *sprite*.

« Le logiciel Scratch ne permet pas de connecter une manette de jeu par USB. Nous allons dépasser cette limitation et produire notre propre manette de jeu. Pour ce faire, nous allons assembler des pièces en bois et du fil électrique afin de construire un système de pédale. Dans la dernière partie de l'atelier nous verrons comment cette pédale peut être associée à une carte électronique programmable pour contrôler des *sprites* dans le jeu vidéo. »

- L'animateur présente les éléments qui seront utilisés pour monter la pédale :

« Nous allons utiliser :

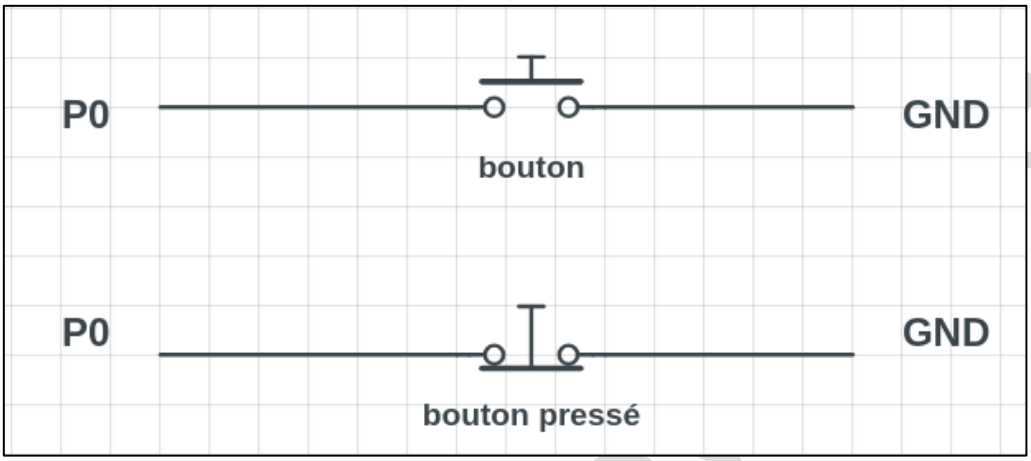
- 7 pièces en bois. Dans l'annexe 5, vous trouverez les fichiers qui permettent de produire ces pièces avec une découpeuse laser.
- 3 mètres de fil électrique.
- Une charnière (en métal ou imprimée en 3D) et un ressort. »

ℹ *Tous ces éléments peuvent être surcyclés, c'est-à-dire récupérés sur d'anciens objets.*

Version provisoire

- Il explique ensuite :

« La pédale que nous allons construire fonctionnera comme un bouton poussoir. Lorsque le joueur enfoncera la pédale jusqu'au socle, les câbles électriques situés sur ces deux objets entreront en contact. Ce contact permettra de fermer le circuit électrique (= bouton enfoncé). Lorsque le joueur relâchera la pression, le ressort permettra à la pédale de reprendre sa position haute et d'ouvrir le circuit électrique (= bouton relâché).



- L'animateur invite les participants à monter leur système de pédale en suivant les instructions de l'Annexe 4.

4.4. Utiliser la pédale comme un émulateur de clavier

\\ Niveau : débutant à intermédiaire

\\ Concepts & notions : émulateur, résistance de rappel (pull-up), état haut/bas.

\\ Modalité : L'animateur illustre la tâche. Les participants répètent la tâche par groupes de deux.

- L'animateur poursuit :

« Un clavier d'ordinateur est une interface homme-machine munie de touches permettant à l'utilisateur d'entrer des lettres ou des chiffres dans l'ordinateur. Chacune de ses touches est un bouton poussoir. Notre objectif est d'émuler (simuler) le signal produit par une de ces touches à l'aide de notre pédale. »

- L'animateur explique et montre :

« Nous allons utiliser une carte électronique programmable appelée micro:bit pour :

- envoyer un signal électrique dans le circuit électrique de la pédale.

- détecter l'ouverture/fermeture du circuit électrique de la pédale.

- générer un signal équivalent à celui produit lorsqu'on appuie sur la touche "h" d'un clavier.

Version pro

- transmettre ce signal à l'ordinateur, la tablette ou le smartphone par Bluetooth. »

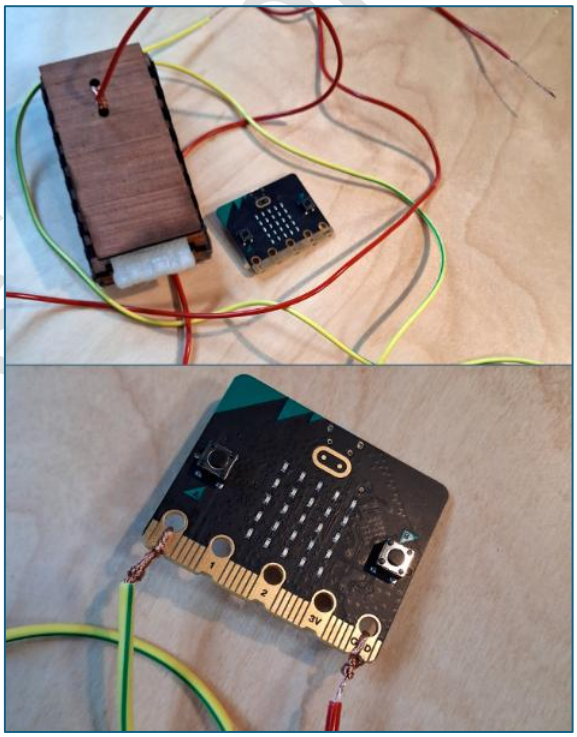
- L'animateur projette le programme de l'annexe 6 et explique brièvement la fonction du programme.

Le programme utilisé sur la carte micro:bit a été édité sous forme de blocs dans MakeCode. L'extension 5.1 permet d'appréhender ce programme dans le détail.

- L'animateur distribue une carte électronique à chaque participant et explique que chacune des cartes à un nom composé de 5 lettres.

Nous recommandons d'inscrire le nom [xxxxx]de chaque carte micro :bit sur cette dernière avant l'activité. Cela évitera une confusion certaine lorsque les participants devront connecter leur carte à un PC ou une tablette par le Bluetooth.

- L'animateur illustre alors comment connecter la pédale à la carte micro:bit.



- Il demande aux participants de charger le fichier de l'annexe 7 dans le logiciel Scratch. Ensuite, il explique :

Version pro

« Prenez quelques minutes pour analyser le programme. Il est un peu plus complexe que celui que vous avez créé mais vous devriez pouvoir le lire et le comprendre. Ici, il est important de noter qu'il y a trois *sprites* différents et que chacun d'entre eux a son propre programme. »

ℹ *Les conditions utilisées pour le sprite de la voiture contiennent l'opérateur 'et' ce qui signifie que les éléments énoncés de part et d'autre du 'et' doivent être vrais pour que la condition soit remplie et que la section de code associée soit exécutée.*

- L'animateur propose aux participants de connecter la carte micro :bit à leur ordinateur ou leur tablette par Bluetooth puis de tester le système.
- Il ajoute :

« Remarquez aussi que lorsque vous appuyez sur 'h', la voiture semble accélérer. En réalité, la voiture n'avance pas, c'est le décor (*sprites* représentant la route) qui descend. On appelle cet effet le *scrolling*. Le *scrolling* est très utilisé dans les jeux vidéo. »

Version provisoire

5. Extensions

5.1. Comment simuler un clavier avec une carte électronique micro:bit et un circuit électrique pour interagir avec un ordinateur ?

Introduction

Lors de l'atelier, vous avez :

1. Appris qu'un clavier d'ordinateur est une interface homme-machine munie de touches permettant à l'utilisateur d'entrer des lettres ou des chiffres dans l'ordinateur. Chacune de ses touches est un bouton poussoir.
2. Connecté le circuit électrique de votre pédale à une carte micro:bit capable de déterminer si le circuit électrique est ouvert ou fermé et d'envoyer cette information vers l'ordinateur ou la tablette.
3. Été informé que la carte électronique doit être programmée :
   - pour simuler le signal produit par une touche de clavier
   - envoyer les informations à l'ordinateur par le Bluetooth.

Objectif de cette extension

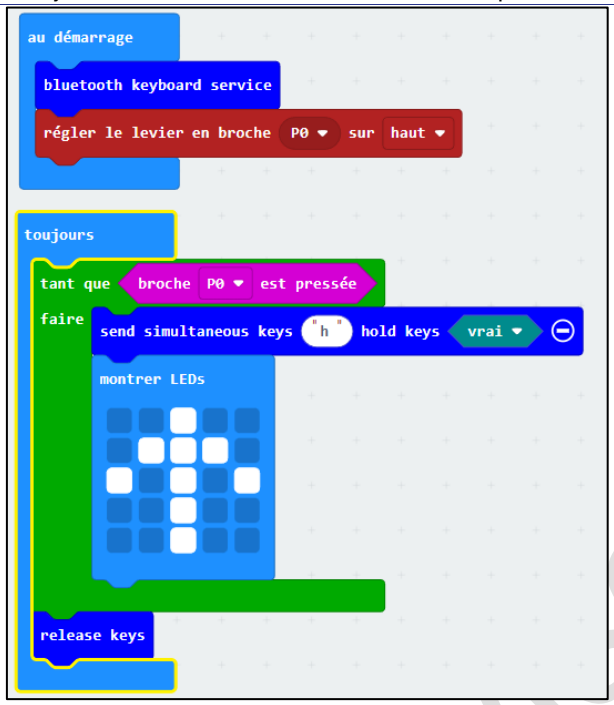
Expliciter le programme utilisé sur la carte électronique micro:bit afin que vous puissiez l'adapter à vos besoins.

Description du programme

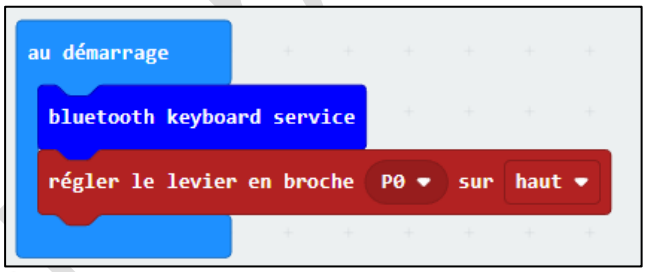
La carte micro:bit peut être programmée dans MakeCode un logiciel gratuit qui permet de programmer par blocs (comme dans Scratch !). Le programme (Annexe 7) ci-dessous a été téléchargé sur les cartes utilisées en atelier. Vous pouvez le téléverser dans MakeCode pour l'étudier ou le modifier.

ℹ Il est recommandé d'utiliser les navigateurs Chrome ou Edge pour programmer dans MakeCode. Certaines fonctionnalités n'apparaissent pas dans Firefox et Safari, par exemples.

Version provisoire



Dans le bloc « au démarrage », il y a deux blocs « Bluetooth keyboard service » et « régler la broche P0 sur haut ». Ces lignes de codes sont lues une fois dès que la carte est alimentée en électricité. Ils initialisent le service Bluetooth et déterminent l'état initial de la broche (aussi appelée pin) P0 comme étant haut.

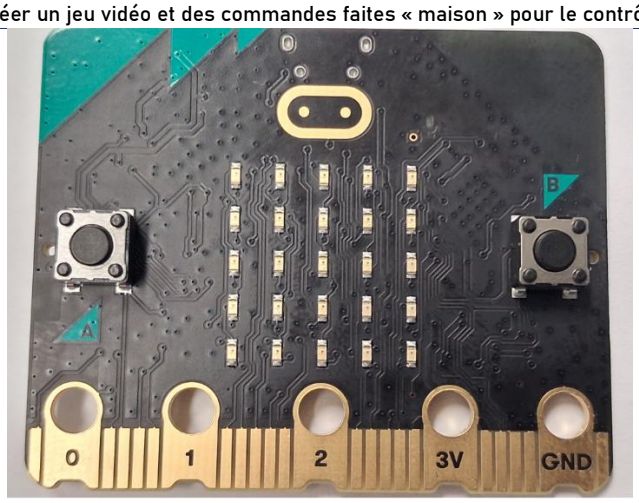


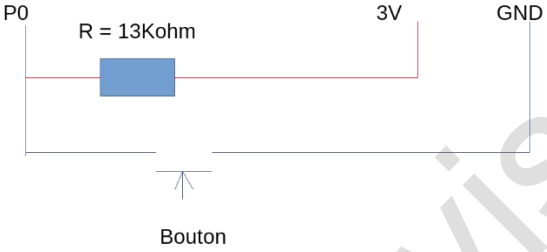
La broche P0 de la carte micro:bit est connectée à la source d'électricité de la carte (3V) par l'intermédiaire d'un fil de connexion interne à la carte.

- Lorsque le circuit externe associé à la broche P0 est ouvert (bouton relâché), le fil de connexion interne assure le passage de courant de la broche 3V vers la broche P0. Cela permet d'assurer une différence de potentiel supérieur à 1,8 V au niveau de la broche P0, soit un état haut (=1). On parle de résistance de rappel (pull-up).

- Lorsque le circuit externe associé à la broche P0 est fermé (bouton enfoncé), le courant passe par le circuit externe (vers le GND) et le voltage mesuré à la broche P0 est inférieur à 1,8 V, soit un état bas (=0).

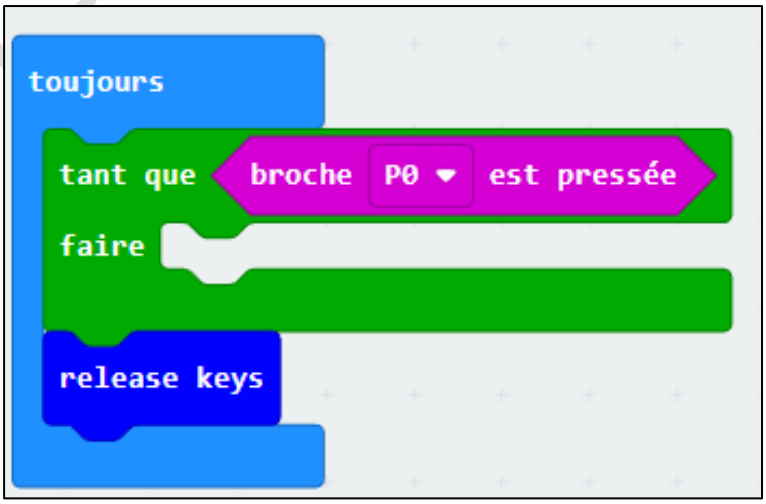
Version provisoire





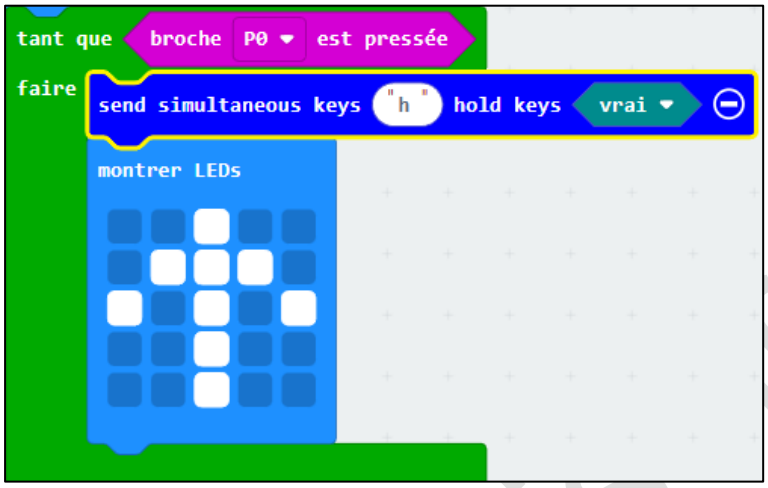
De plus, le fil interne est équipé d'une résistance de 13 kΩ. Lorsque le bouton poussoir est enfoncé, la résistance réduit le flux de courant vers le circuit externe et la broche GND pour éviter les courts-circuits.

La série de blocs placés dans le bloc « toujours », les blocs « tant que … » et « release » seront lus successivement, à l'infini. Cependant, le passage du bloc « tant que … » au bloc « release » ne se produit pas tant que la condition « broche P0 est pressée » est vérifiée, c'est-à-dire tant que le bouton poussoir reste enfoncé. En programmation, on parle d'une boucle 'while' ou while loop.



Version provisoire

Les deux blocs présents dans le bloc « tant que… » permettent d'envoyer un signal équivalent à celui de la lettre 'h' d'un clavier et d'allumer certains LEDs de la carte micro:bit de façon à représenter une flèche pointant vers le haut.



Version provisoire

5.2. Ajouter une variable dans le programme pour collecter des infos sur les déplacements de la voiture.

Introduction

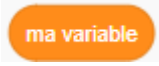
Lors de l'atelier, vous avez appris à utiliser plusieurs concepts de base en programmation, dont la boucle et la condition. Il vous reste à appréhender un autre pilier des principes fondamentaux de la programmation, la variable. La maîtrise de ce concept ouvre le champ de possibles en programmation.

Objectif de l'extension

Intégrer le concept de variable au jeu vidéo initié lors de l'atelier.

Exercices pour comprendre et intégrer la notion de variable

\\ Charger le programme de l'annexe 3 dans Scratch.

1. Créer un compteur pour 

savoir combien de fois la voiture traverse la moitié inférieure de l'écran.

ℹ *Créer une variable et lui donner un nom explicite (ex : « compteur_trajet »).*

ℹ *Au début du programme, attribuer une valeur initiale à la variable.*

ℹ *Changer la valeur associée à cette variable à chaque fois qu'un trajet est réalisé.*

2. Une autre variable était déjà présente dans le programme de l'annexe 3. De quel bloc s'agit ? Quelle information est renvoyée par cette variable ?

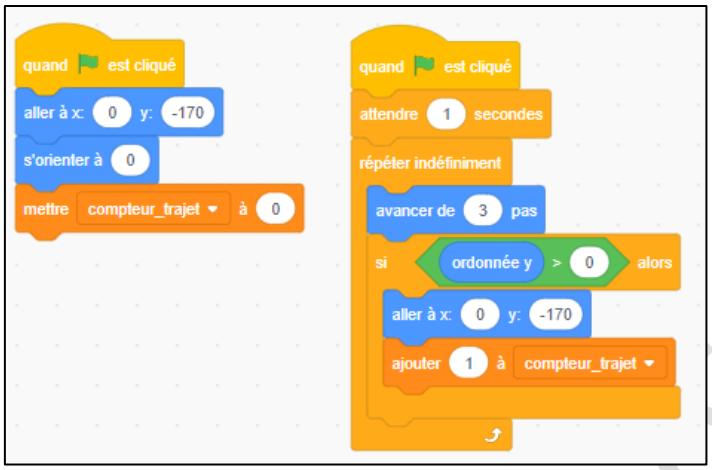
ℹ *Une fois de plus, la forme des blocs est utile.*

Vous venez de programmer une **variable** !
En informatique, les variables sont des espaces mémoire "élémentaires" qui associent un nom (l'identifiant) à une valeur (https://fr.wikipedia.org/wiki/Variable_(informatique)). En fonction du type de variable, cette valeur peut être un nombre, du texte ou d'autres données. Cette valeur peut être lue ou modifiée par le programme pendant son exécution.

Version provisoire

Solutions des exercices

1. Exemple de solution



2. Solution : Il s'agit de la variable « ordonnée y » qui renvoie la position de la voiture sur l'axe Y.

Version provisoire

5.3. Utiliser l'accéléromètre de la carte électronique pour diriger la voiture

Rappel

La carte micro:bit peut être utilisée pour envoyer un signal équivalent au signal d'une touche de clavier vers un ordinateur ou une tablette. Lors de l'atelier, vous avez utilisé la carte micro:bit pour envoyer un signal ('lettre h') lorsque la pédale est enfoncée. La carte micro:bit est équipée d'un accéléromètre renvoyant de valeurs allant de -1024 à +1024 en fonction de l'inclinaison de la carte. Il est donc possible d'envoyer à un ordinateur des signaux différents en fonction des valeurs mesurées par l'accéléromètre de la carte.

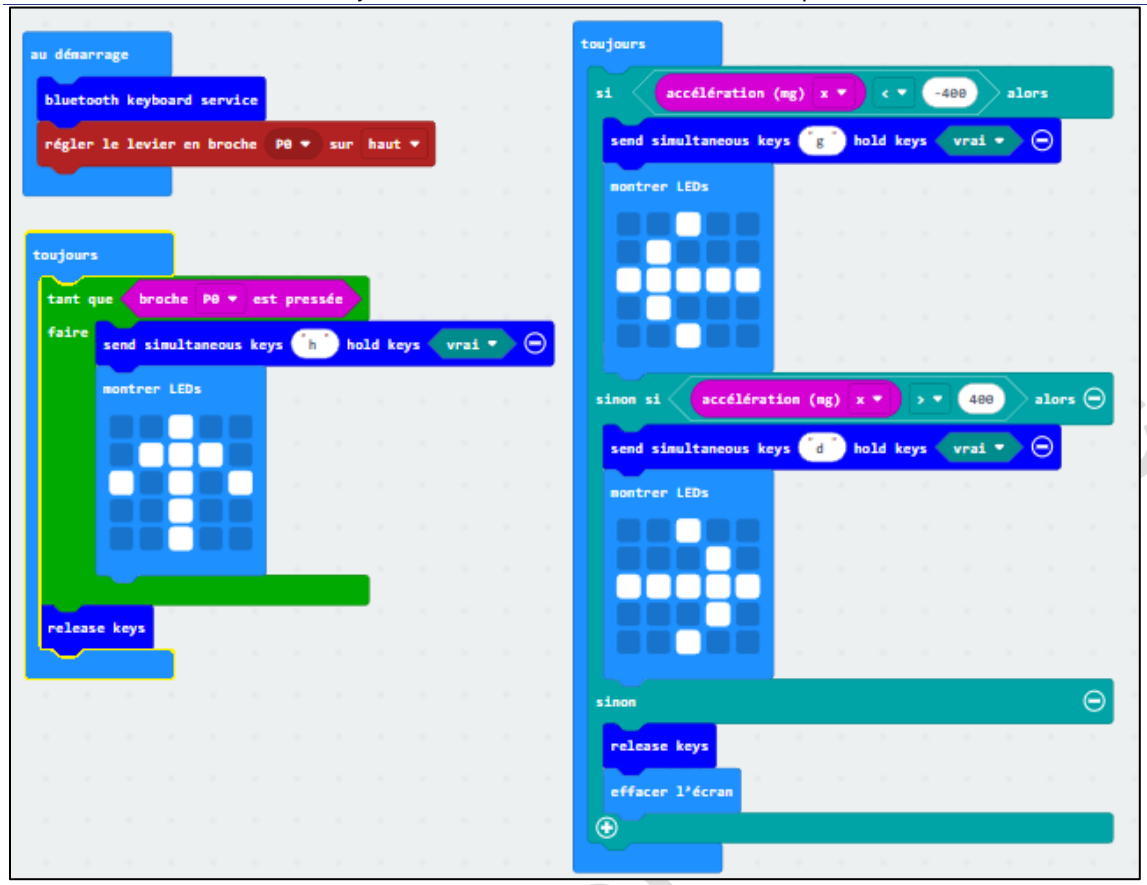
Objectif de l'extension

Produire un programme qui permet de transférer l'inclinaison gauche/droite d'un volant vers l'ordinateur ou la tablette.

Description du programme

Par rapport au programme MakeCode de l'extension 5.1, le programme MakeCode (Annexe 8) présenté ci-dessous contient une boucle « toujours » supplémentaire. Cette fois, la valeur d'entrée « accélération (mg) » doit être inférieure à -400 pour que le programme renvoie le signal de la lettre 'g' à l'ordinateur ou la tablette. Si cette valeur est supérieure à 400, c'est le signal de la lettre 'd' qui sera émis par la carte micro:bit. Pour des valeurs d'inclinaison entre -400 et 400, la carte n'envoie pas de signal.

Version provisoire



La carte peut être placée sur un volant en bois comme celui de la photo ci-dessous. Ce dernier peut être découpé au laser ou imprimer en 3D.

Version pr

5.4. Programmer la version « complète » du jeu.

Rappel

Lors de l'atelier vous avez programmé les prémices d'un jeu vidéo dans Scratch. Ce jeu peut être sans cesse enrichit et modifié. Lorsqu'un programme devient complexe, il est important de le commenter afin de pouvoir le partager avec d'autres concepteurs ou en vue de modifications futures.

Objectif de l'extension

1/Proposer une version plus avancée du jeu vidéo de l'atelier.

2/Illustrer l'intérêt de commenter son programme.

Description du programme

Les blocs sont commentés directement dans le programme Scratch (Annexe 9).

Version provisoire