

Domotique avec Arduino et Scratch

Une séquence du projet *1,2,3... CODEZ !*

Résumé

Cette séquence s'adresse en priorité à des élèves débutants en programmation Scratch et en robotique basée sur Arduino. L'objectif est de fabriquer une maquette de maison domotique : différents capteurs et actionneurs, installés dans la maquette, sont pilotés par une carte Arduino. La programmation se fait à l'aide de mBlock (une extension de Scratch permettant de piloter Arduino). Ces activités nécessitent un équipement particulier, aussi bien matériel que logiciel : plusieurs options sont possibles.

Projet « Domotique avec Arduino »

Ce projet s'adresse en priorité à des élèves débutants en programmation *Scratch* et en robotique basée sur *Arduino*.

Discipline concernée et liens avec les programmes

Les programmes de technologie de 2016 se prêtent particulièrement bien à un tel projet, qui permet d'aborder des notions d'algorithmique, de programmation et de robotique, en particulier sur le cas de la domotique.

Nous ne citons, ci-dessous, que les connaissances et compétences explicitement travaillées dans ce projet, sans mentionner les compétences transversales. Dans un souci de synthèse, nous évitons les redondances et ne citons que ce qui est dans la rubrique « informatique et programmation ».

L'informatique et la programmation

- Écrire, mettre au point et exécuter un programme
- Analyser le comportement attendu d'un système réel et décomposer le problème posé en sous-problèmes afin de structurer un programme de commande.
- Écrire, mettre au point (tester, corriger) et exécuter un programme commandant un système réel et vérifier le comportement attendu.
- Écrire un programme dans lequel des actions sont déclenchées par des événements extérieurs.
 - Notions d'algorithme et de programme.
 - Déclenchement d'une action par un événement, séquences d'instructions, boucles, instructions conditionnelles.
 - Systèmes embarqués.
 - Forme et transmission du signal (analogique / numérique).
 - Capteur, actionneur, interface.
- Observer et décrire le comportement d'un robot ou d'un système embarqué. En décrire les éléments de sa programmation.
- Agencer un robot (capteurs, actionneurs) pour répondre à une activité et un programme donnés.
- Écrire, à partir d'un cahier des charges de fonctionnement, un programme afin de commander un système ou un système programmable de la vie courante, identifier les variables d'entrée et de sortie.
- Autres compétences :
- Développer les bonnes pratiques de l'usage des objets communicants
- Logiciels de *mind-mapping*.

Préparation du projet



Ce projet nécessite un équipement particulier, aussi bien matériel que logiciel. Plusieurs options sont possibles. Ces options sont détaillées dans l'avant-propos de la Séance 3, pages 204 et suivantes.




Objectifs

L'objectif du projet est de fabriquer une maquette de maison domotique: différents capteurs et actionneurs, installés dans la maquette, sont pilotés par une carte *Arduino*. La programmation se fait à l'aide de *mBlock* (une extension de *Scratch* permettant de piloter *Arduino*).

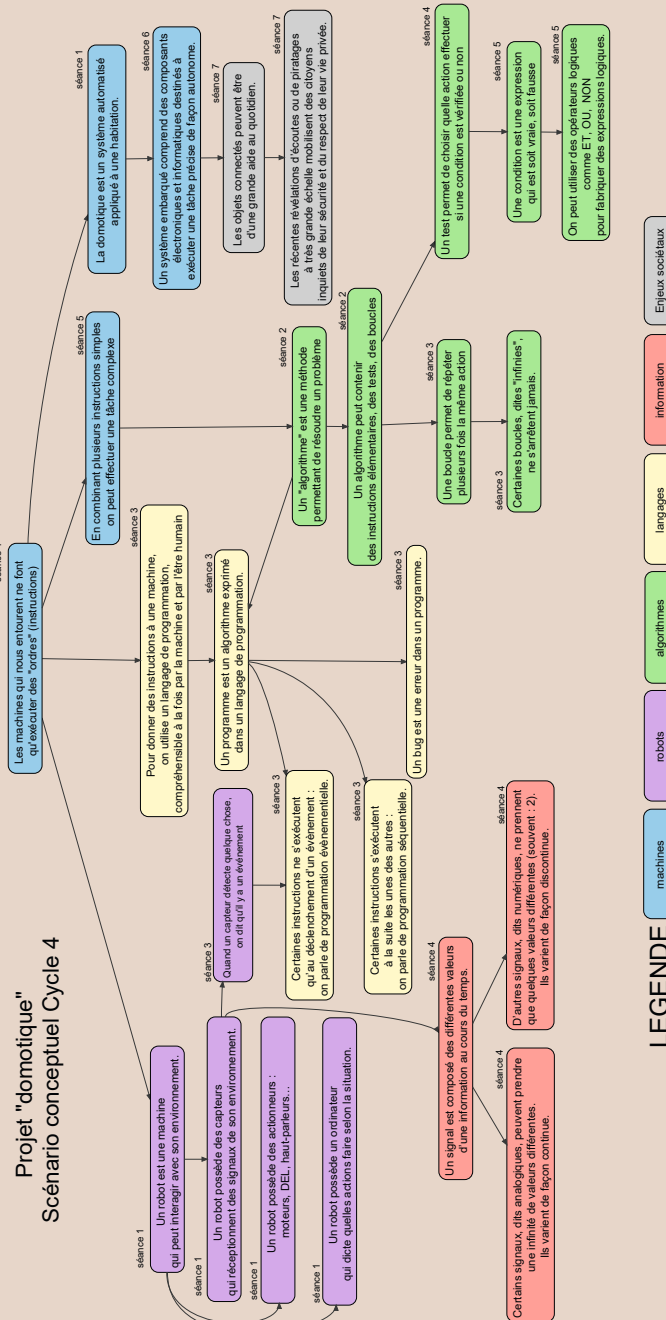
Les élèves travaillent sur la façon d'assurer la sécurité des biens et des personnes dans une maison, établissent la liste des fonctions et trouvent des solutions techniques pour chacune (détection de mouvement, d'ouverture ou de fumée, simulation de présence, bouton « appel d'urgence »...).

Ils apprennent à brancher des capteurs et actionneurs sur une carte *Arduino*, et à programmer l'ensemble sur *mBlock*. Ils apprennent à différencier un signal analogique d'un signal numérique, et à utiliser des opérateurs logiques pour effectuer des fonctions complexes. Après la réalisation de leur maquette de maison domotique, les élèves débattent des enjeux sociétaux de la domotique et, plus généralement, des objets connectés. Le projet s'achève par une restitution du travail réalisé.

	Séance	Titre	Page	Résumé
	Séance 1	Comment sécuriser une maison ?	196	Les élèves s'interrogent sur les différents moyens permettant de sécuriser une maison, qu'il s'agisse de la protection des biens ou des personnes (alarme, détection d'ouverture, de présence, de fumée, de chute, etc.). Ils découvrent qu'en domotique, on automatise des fonctions de l'habitat à l'aide d'un ordinateur équipé d'un programme. L'ordinateur reçoit les informations de capteurs et commande des actionneurs.
	Séance 2	Solutions techniques	200	Chaque groupe recense les fonctions (simuler une présence, signaler la présence d'une personne...), établit la ou les solutions techniques pour certaines d'entre elles... et présente ces solutions à la classe entière. Les élèves s'initient à la notion d'algorithme. En parallèle des séances suivantes, ils peuvent commencer à fabriquer la maquette de la maison.
	Séance 3	Introduction à <i>Scratch</i> et à <i>Arduino</i>	204	Les élèves découvrent comment commander l'allumage et l'extinction d'une DEL à l'aide de la carte <i>Arduino</i> . Pour cela, ils réalisent le montage électrique de la DEL, écrivent le programme à l'aide du logiciel <i>mBlock</i> et chargent ce programme sur la carte <i>Arduino</i> .
	Séance 4	Étude de quelques capteurs: analogiques ou numériques ?	216	Les élèves étudient différents capteurs à l'aide de montages très simples: modalités de branchement, d'interrogation et de contrôle, valeurs renvoyées par ces capteurs. Ils distinguent alors les signaux analogiques et numériques. Enfin, ils apprennent à réaliser un montage plus complet permettant au microcontrôleur de piloter un actionneur à partir des informations fournies par un capteur.
	Séance 5	Comportements complexes	223	Les élèves apprennent à manipuler des opérateurs logiques, en particulier ET et OU, de façon à pouvoir programmer des comportements plus complexes, en lien avec les cahiers des charges produits en début de projet. Ils réalisent les montages et les programmes correspondant. NB: cette séance peut être dédoublée si les élèves ont des difficultés à manipuler les opérateurs logiques (cf. activité de prolongement conseillée).

	Séance 6	Fabrication de la maquette	229	Les élèves fabriquent ou finalisent leurs maquettes incluant les éléments physiques de la maison et les solutions domotiques (carte <i>Arduino</i> , capteurs, actionneurs, câbles). Pour permettre à cette maquette de fonctionner indépendamment de l'ordinateur, et ainsi réaliser un vrai système embarqué, ils apprennent à utiliser le mode « autonome » d' <i>Arduino</i> .
	Séance 7	Débat philo : les enjeux sociétaux de la domotique	234	Les élèves participent à un « atelier philo » portant sur les enjeux actuels de la domotique, et plus généralement des objets connectés. Facilitent-ils vraiment la vie des utilisateurs ? Permettent-ils de réduire l'isolement de chacun ? Présentent-ils des risques pour la sécurité ?
	Séance 8	Restitution du projet	240	Les élèves réalisent un diaporama (ou une vidéo de type <i>making of</i>) destiné à présenter leur projet (aux autres professeurs, aux autres classes, aux familles...).

Scénario conceptuel





Séance 1 – Comment sécuriser une maison ?

Discipline dominante	Technologie
Résumé	Les élèves s'interrogent sur les différents moyens permettant de sécuriser une maison, qu'il s'agisse de la protection des biens ou des personnes (alarme, détection d'ouverture, de présence, de fumée, de chute, etc.). Ils découvrent qu'en domotique, on automatise des fonctions de l'habitat à l'aide d'un ordinateur équipé d'un programme. L'ordinateur reçoit les informations de capteurs et commande des actionneurs.
Notions (cf. scénario conceptuel, page 195)	<p>Machines :</p> <ul style="list-style-type: none"> • Les machines qui nous entourent ne font qu'exécuter des « ordres » (instructions). • La domotique est un système automatisé appliqué à une habitation. <p>Robots :</p> <ul style="list-style-type: none"> • Un robot est une machine qui peut interagir avec son environnement. • Un robot possède des capteurs qui réceptionnent des signaux de son environnement. • Un robot possède des actionneurs : moteurs, DEL, haut-parleurs... • Un robot possède un ordinateur qui dicte quelles actions faire selon la situation.
Type d'activité	Séance débranchée
Matériel	<p>Pour chaque groupe (îlot) :</p> <ul style="list-style-type: none"> • Photocopie de la Fiche 1 (page 199) • Un appareil photo numérique

Situation déclenchante

Le professeur introduit une situation déclenchante du type « la famille Dupond souhaite construire une maison dans un lieu isolé, comment peuvent-ils l'équiper pour être en sécurité ? ».

Il peut également évoquer une évolution récente de l'habitat : les logements utilisent de plus en plus d'outils numériques et sont de plus en plus connectés. Il demande aux élèves comment l'on pourrait profiter de ces nouvelles technologies pour améliorer la sécurité dans un logement.

Avant de les laisser travailler en groupe, il leur annonce que ce projet se terminera par une restitution finale, pour laquelle ils devront présenter les différentes étapes de leur projet : leurs idées initiales, leurs réalisations... Cette restitution peut prendre plusieurs formes différentes (diaporama, vidéo de type making of, etc.). Dans cette optique, il les invite à prendre des photos de ce qu'ils font et de ce qu'ils écrivent, et à tenir un « cahier de bord » ou « cahier de projet ». La structure de ce cahier peut être imposée par le professeur, ou discutée en classe.

Exemple de structure :

Date	Tâche – activité	Production	Modalités	Moyens
	<i>Ce que nous devons faire</i>	<i>Présentation orale, affiche, fichier, maquette...</i>	<i>Seul – en binôme – en groupe...</i>	<i>Ordinateur, outils, machines, documentation...</i>

Recherche (par groupes)

Les élèves sont répartis en groupes (de 4 à 7 élèves en fonction des effectifs, l'objectif étant d'avoir 5 à 6 groupes dans la classe) qui constitueront des « îlots » stables pour toute la durée du projet. Chaque groupe réfléchit aux différents moyens de sécuriser un logement.

Après quelques minutes de recherche, le professeur distribue à chaque groupe une photocopie de la Fiche 1, qui présente le plan d'une maison très simple¹.

Les élèves doivent reporter, sur ce plan, les différents éléments auxquels ils ont pensé, par exemple :

Note pédagogique

Il est utile de prévoir, dès le début de chaque phase de recherche, quel élève sera le « secrétaire » du groupe, chargé d'écrire les documents de présentation, et lequel sera le « porte-parole », qui présentera les travaux du groupe lors des mises en commun. Le professeur veillera à ce que ces deux rôles tournent régulièrement au sein des groupes. Le cahier de bord pourra être utile pour cela.

Mise en commun (collectivement)

Le professeur organise une mise en commun au cours de laquelle chaque groupe présente les idées notées sur le plan.

La discussion collective fait émerger le fait que la sécurité d'une maison peut être traitée selon deux approches complémentaires :

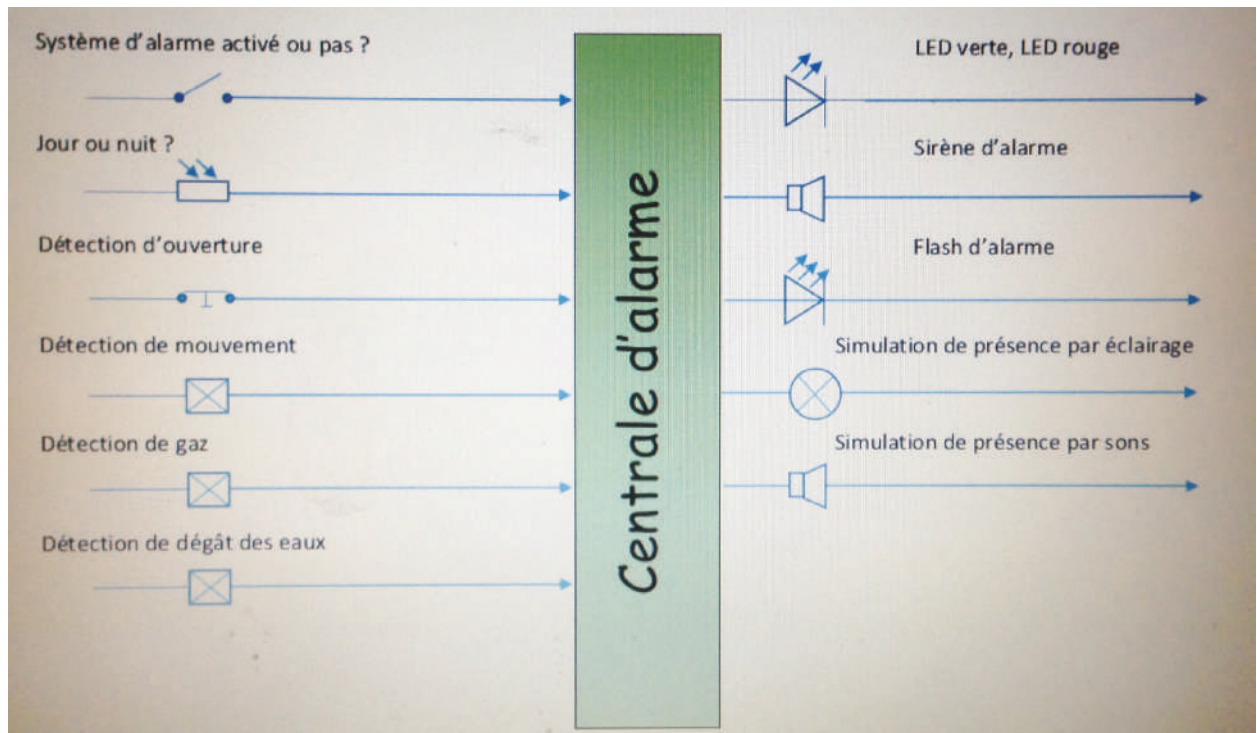
- **La sécurité des biens.** Il s'agira, pour l'essentiel, de se prémunir contre le vol. La domotique permettra de détecter des intrusions (mouvement, ouverture de porte, choc...), de déclencher une alarme (lumière, sirène), voire de décourager d'éventuels intrus en simulant une présence lorsque les habitants de la maison sont absents (lumière, musique...). On peut compléter le dispositif par un envoi de message (au propriétaire ou à une agence de sécurité).
- **La sécurité des personnes.** Il s'agira, ici, de prémunir les habitants de la maison contre certains risques domestiques mettant leur vie en danger. On peut penser à la détection des gaz toxiques ou des fumées, ou à la programmation d'un bouton « appel d'urgence » (qui peut être utile aux personnes âgées par exemple).

Au cours de cette mise en commun, le professeur introduit le vocabulaire spécifique en guidant les élèves par un questionnement :

- Qu'est-ce qu'un système automatisé ? De quoi est-il composé ? L'objectif est d'introduire la distinction entre un « capteur » (qui détecte un signal) et un « actionneur » (qui effectue une action). Les élèves comprennent facilement la différence entre capteurs et actionneurs, mais oublient souvent qu'un troisième élément est indispensable pour relier les deux : un ordinateur, équipé d'un programme. C'est le programme qui permet à l'ordinateur de déclencher des actions lorsqu'un ou plusieurs capteur(s) envoie(nt) un signal (ce signal véhiculant une information).
- Que signifie le mot « domotique » ? Il est formé par la contraction de deux termes ? Lesquels ? Que signifie « *domus* » ? L'objectif étant de définir la domotique comme un cas particulier de système automatisé appliqué à une habitation.

1. Ce plan peut bien sûr être modifié. Le fichier source (éditable à l'aide du logiciel *Sweet Home 3D*) est téléchargeable sur le site Web dédié au projet (cf. page 404).

Au besoin, la classe réalise collectivement une carte mentale ou un schéma résumant le projet de maison domotique. Exemple de schéma (incomplet car en cours de réalisation):



Conclusion

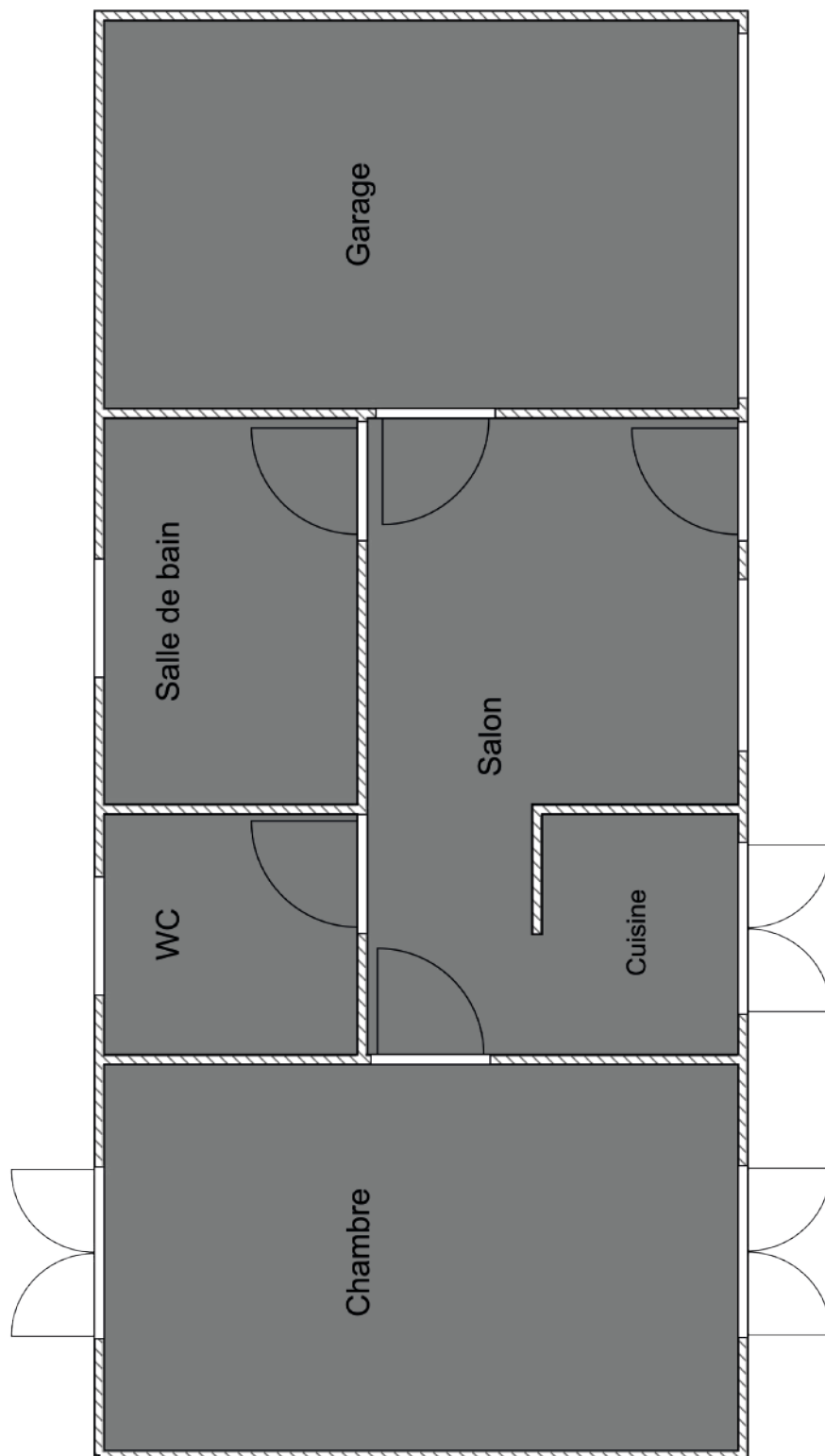
La classe définit collectivement les mots-clés cités ci-dessus : capteur, actionneur, programme, signal, domotique. Elle finalise la liste des fonctions qui ont été évoquées lors de la mise en commun :

- Simuler une présence
 - Allumer des lumières (uniquement s'il fait nuit) et faire du bruit
- Détecter les intrusions
 - Programmer une alarme (sirène + lumière) déclenchée par :
 - Le mouvement
 - L'ouverture d'une porte ou fenêtre
 - Un choc sur une porte ou fenêtre
- Prévenir contre les risques domestiques
 - Détecter les fumées et gaz toxiques
 - Allumer la lumière lorsqu'un mouvement est détecté (la nuit uniquement)
 - Programmer un bouton « appel d'urgence »

Les élèves archivent les notes et photos qu'ils ont prises dans le but de préparer la restitution finale.

FICHE 1 Sécuriser une maison

Consigne: Indique sur le plan l'emplacement des différents capteurs ou actionneurs te permettant de sécuriser cette maison.





Séance 2 – Solutions techniques

Discipline dominante	Technologie
Résumé	Chaque groupe recense les fonctions (simuler une présence, signaler la présence d'une personne...), établit la ou les solutions techniques pour certaines d'entre elles... et présente ces solutions à la classe entière. Les élèves s'initient à la notion d'algorithme. En parallèle des séances suivantes, ils peuvent commencer à fabriquer la maquette de la maison.
Notions (cf. scénario conceptuel, page 195)	Algorithmes : <ul style="list-style-type: none">• Un algorithme est une méthode permettant de résoudre un problème.• Un algorithme peut contenir des instructions élémentaires, des tests, des boucles.
Type d'activité	Séance débranchée
Matériel	Pour chaque groupe : <ul style="list-style-type: none">• Un ordinateur équipé d'un logiciel de diaporama (<i>OpenOffice Presentation, MS Powerpoint...</i>)• Un appareil photo numérique• (facultatif) des « cartes logigrammes » prédécoupées (cf. corps de la séance)

Situation déclenchante

Le professeur explique que la réalisation d'un système technique nécessite dans un premier temps de lister les fonctions, et dans un second temps de décrire comment mettre en œuvre ces fonctions : les « solutions techniques ». Il propose aux élèves un document-type² afin de les guider dans la présentation de leurs solutions techniques, et afin de faciliter les mises en commun :

- Fonction
- Solution trouvée
- Capteur(s) nécessaire(s)
- Actionneur(s) nécessaire(s)
- Algorithme

Les fonctions possibles sont : simuler une présence, signaler la présence d'un individu, signaler une tentative d'intrusion, signaler un dégagement anormal de fumée ou de gaz, appeler les secours...

Recherche (par groupes)

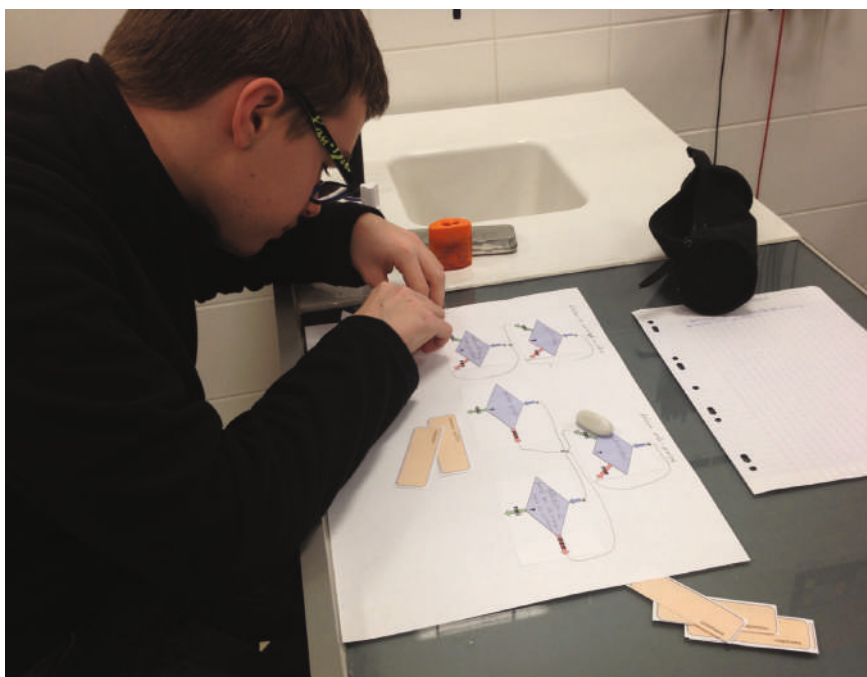
Les élèves (répartis dans les mêmes groupes que précédemment) choisissent de traiter une fonction. Ils définissent la solution technique et remplissent le document de présentation, en respectant la forme choisie par l'enseignant.

Les groupes ayant terminé en avance peuvent travailler sur d'autres fonctions.

2. Nous proposons un exemple modifiable sur le site Web du projet.

Notes pédagogiques

- Il est préférable de s'arranger pour que chaque fonction soit étudiée par deux groupes différents, de manière à pouvoir comparer les différentes propositions lors de la mise en commun.
- Si les élèves n'ont jamais réalisé de programme auparavant, il n'est pas nécessaire (à ce stade) de leur demander de formaliser un algorithme sous la forme de logigramme (ce sera fait plus tard dans cette séquence). L'algorithme peut simplement être décrit à l'aide de phrases, car par définition il s'agit d'une méthode pour résoudre un problème: il n'y a pas de représentation meilleure qu'une autre. Le français s'y prête très bien, mais on peut également le représenter de façon graphique (logigrammes) ou l'implémenter dans un langage de programmation (ce qui constituera alors un « programme »).
- Si l'enseignant le souhaite, il peut distribuer aux élèves des cartes vierges permettant de réaliser les logigrammes, comme sur la photo ci-dessous. La construction d'un logigramme ne présente pas de difficulté particulière, néanmoins, il peut être utile de réaliser collectivement les premières étapes, avant de laisser les élèves en autonomie.



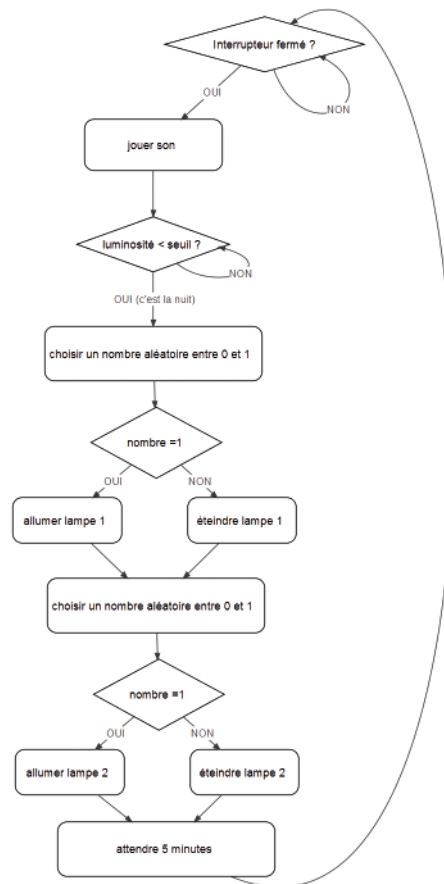
Exemple de réalisation d'un logigramme à partir de cartes vierges prédécoupées.
Classe de Serge Sauton et Willy Bouichou (Beauvais).

Mise en commun (collectivement)

Le porte-parole de chaque groupe vient présenter sa solution technique. La discussion collective et la comparaison des différentes propositions permettent de valider, pour chaque fonction, le principe général, le matériel nécessaire, ainsi que le détail de l'algorithme.

Exemple de présentation de solution technique pour la fonction « simuler une présence » (la plus complexe, car elle fait intervenir plusieurs capteurs, plusieurs actionneurs, ainsi que des tirages aléatoires et une temporisation). Ici, deux modes d'expression de l'algorithme sont présentés: en français et sous forme de logigramme.

- Fonction : simuler une présence dans l'habitation de manière à décourager les tentatives d'effraction
- Solution trouvée : en cas d'absence du résident, activer un simulateur qui va allumer différentes lumières aléatoirement et jouer de la musique
- Capteurs nécessaires (type et quantité)
 - 1 interrupteur (activation du mode « simulation de présence »)
 - 1 capteur de luminosité (permettant de savoir s'il fait jour ou nuit)
- Actionneurs nécessaires (type et quantité)
 - 2 lumières de type DEL, de couleur blanche
 - 1 haut-parleur
- Algorithme (exprimé en français)
 - Répéter sans arrêt :
 - Lorsque l'interrupteur est fermé
 - . Jouer un morceau de musique (haut-parleur)
 - . Détecter s'il fait jour ou nuit à l'aide du capteur de luminosité. S'il fait nuit, allumer 1 ou 2 DEL(s) aléatoirement. S'il fait jour, éteindre les DELs.
 - Attendre 5 minutes
- Algorithme (représentation sous forme de logigramme)



Notes pédagogiques

- Cet algorithme n'est qu'un exemple. On peut le raffiner, par exemple en choisissant de déclencher la musique de façon aléatoire, ou en faisant en sorte qu'une des deux lumières soit obligatoirement allumée (ici, on laisse la possibilité que, de temps en temps, les deux lumières soient éteintes alors qu'il fait nuit).

- Cependant, comme nous le verrons à la Séance 5, la programmation de cet algorithme est déjà un beau défi pour les élèves (on proposera même une variante pour le simplifier). Nous déconseillons donc de trop le complexifier.

Traces écrites

Les solutions techniques validées collectivement sont partagées avec l'ensemble de la classe au format électronique (*OpenOffice Presentation, MS Powerpoint...*). Si possible, une version papier est également collée dans le cahier de projet.

Les élèves archivent les notes et photos qu'ils ont prises dans le but de préparer la restitution finale.

Notes pédagogiques

- L'objectif de ce projet est l'apprentissage de la robotique et de la programmation. Nous ne décrivons donc pas la fabrication de la maquette physique de la maison à l'aide de carton plume ou de PVC. L'enseignant peut, s'il le souhaite, faire fabriquer cette maquette par les élèves, ou bien leur proposer une maquette déjà prête afin de passer plus de temps sur ce qui fait l'intérêt du projet : les branchements et les programmes.
- Si ce sont les élèves qui réalisent la maquette de la maison, ils peuvent le faire dès maintenant, en parallèle des séances qui suivent, consacrées à la programmation de la carte *Arduino*, ou bien attendre d'avoir bien compris le fonctionnement des capteurs et actionneurs, afin d'ajuster leur maquette en conséquence. Dans ce cas, la construction de la maquette se déroule entre la Séance 5 et la Séance 6.



Séance 3 – Introduction à *Scratch* et à *Arduino*

Discipline dominante	Technologie
Résumé	Les élèves découvrent comment commander l'allumage et l'extinction d'une DEL à l'aide de la carte <i>Arduino</i> . Pour cela, ils réalisent le montage électrique de la DEL, écrivent le programme à l'aide du logiciel <i>mBlock</i> et chargent ce programme sur la carte <i>Arduino</i> .
Notions (cf. scénario conceptuel, page 195)	<p>Algorithmes :</p> <ul style="list-style-type: none">• Une boucle permet de répéter plusieurs fois la même action.• Certaines boucles, dites « infinies », ne s'arrêtent jamais. <p>Langages :</p> <ul style="list-style-type: none">• Pour donner des instructions à une machine, on utilise un langage de programmation, compréhensible à la fois par la machine et par l'être humain.• Un programme est un algorithme exprimé dans un langage de programmation.• Certaines instructions s'exécutent les unes à la suite des autres : on parle de programmation séquentielle.• Certaines instructions ne s'exécutent qu'au déclenchement d'un événement : on parle de programmation événementielle.• Un bug est une erreur dans un programme.
Type d'activité	Séance branchée (robotique)
Matériel	Pour chaque groupe (cf. dans le corps de la séance pour les différents choix possibles) : <ul style="list-style-type: none">• Un ordinateur sur lequel a été installé le logiciel <i>mBlock</i>• Une carte <i>Arduino</i>, et le câble permettant de la relier au port USB de l'ordinateur• Une pile 9V ou une alimentation pour la carte <i>Arduino</i>• Un <i>Shield Grove</i> (cf. explication dans le corps de la séance)• Une DEL <i>Grove</i>• La Fiche 2, page 214• La Fiche 3, page 215

Notes pédagogiques

- Cette séance a pour but de programmer une carte *Arduino* en *Scratch* (*mBlock*), et non pas de découvrir l'ensemble des fonctionnalités de *Scratch*. Les tâches qui suivent se focalisent donc sur ces fonctionnalités spécifiques. *Scratch* permet de faire beaucoup d'autres choses, comme le montrent les autres projets présentés dans cet ouvrage !
- Le professeur peut mettre en place, avant la présente séance, la séance « Découvrir *Scratch* » extraite de la séquence « programmation d'un jeu d'arcade : initiation à *Scratch* » (page 71). Le temps passé à s'initier aux bases de *Scratch* sera gagné plus tard, les élèves étant plus efficaces et autonomes. Bien entendu, cette séance d'initiation est inutile si les élèves ont déjà manipulé *Scratch* avant, même pour des projets très simples.

Avant-propos : préparer le matériel

Le bon déroulement de cette séance (et des suivantes) nécessite que le professeur ait préparé à l'avance le matériel qui sera utilisé par les élèves.

Carte Arduino

Arduino n'est pas un ordinateur miniature comme peut l'être un *Raspberry Pi* par exemple. Il n'embarque pas de système d'exploitation lui permettant d'être utilisé seul. Il s'agit simplement d'une carte équipée d'un micro-contrôleur capable de piloter des périphériques simples (capteurs, actionneurs), et programmable par l'utilisateur. La programmation doit se faire via un ordinateur.

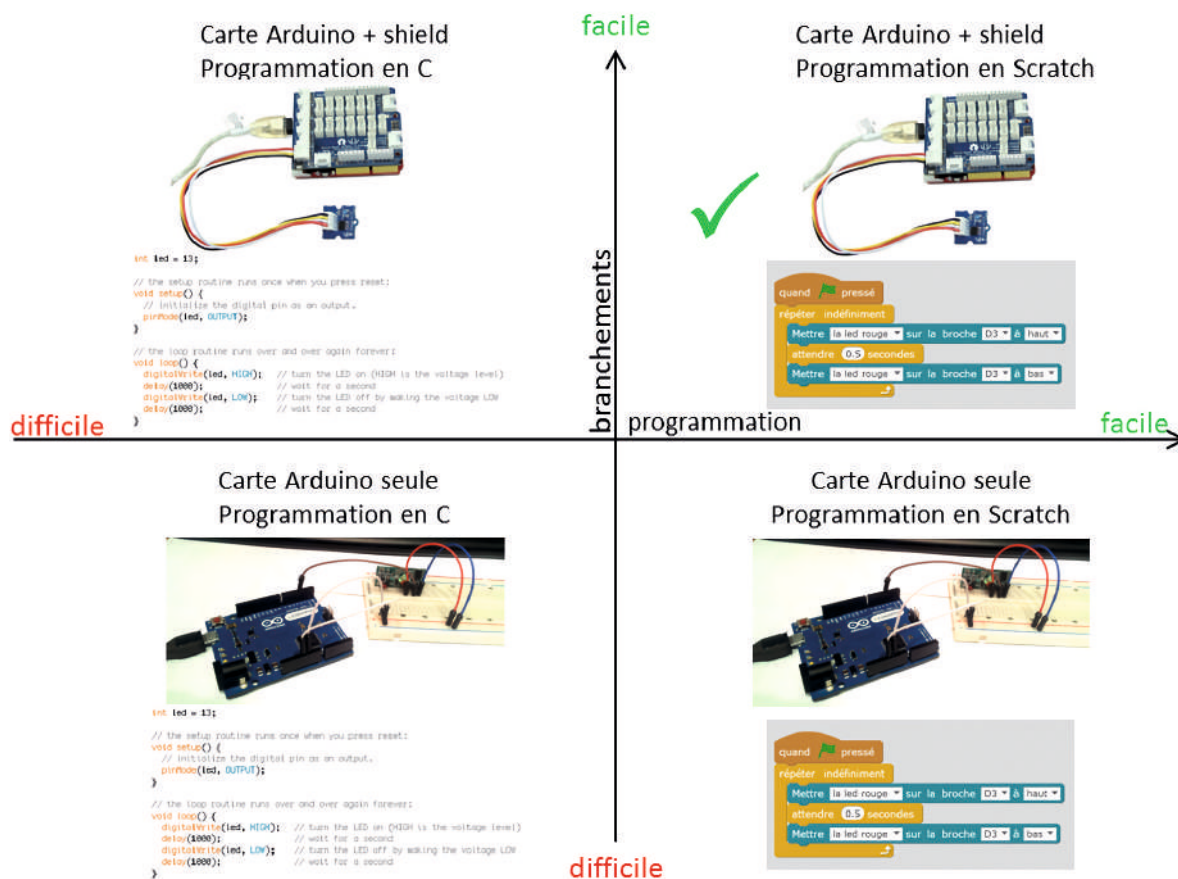
Arduino possède de nombreuses qualités pour l'éducation :

- faible coût;
 - licence libre à la fois pour le matériel et le logiciel;
 - grand nombre de capteurs/actionneurs disponibles sur le marché (eux-mêmes à coût très faible) et grand nombre de bibliothèques de programmation;
 - simplicité d'utilisation;
 - compatibilité multi-plateforme (on peut le programmer à l'aide d'un PC Windows, d'un Mac...);
- communauté très active, en France et dans le monde.

Arduino peut être utilisé de différentes façons :

- matériel
 - carte *Arduino* seule, sur laquelle on branche des capteurs/actionneurs directement. Il faut choisir soi-même les branchements, la polarité, la masse, en prenant bien garde à ajouter des résistances et à ne pas se tromper pour ne pas griller les composants!
 - carte *Arduino* équipée d'un « *Shield* » (bouclier, en français... mais c'est le terme anglais qui est systématiquement employé), qui permet de simplifier considérablement les branchements des capteurs et actionneurs : ils sont branchés par une broche standardisée équipée de détrompeurs.
 - logiciel
 - la carte se programme, de façon native, en langage C. Il s'agit d'un langage de programmation très polyvalent et extrêmement répandu. Cependant, c'est un langage textuel qui nécessite un apprentissage important (lexique, syntaxe...), moins adapté aux élèves de collège.
 - La carte peut également se programmer à l'aide d'un langage graphique (programmation par « blocs ») comme *Scratch*, *Blockly*, *ArduBlock* ou *Snap*. Nous conseillons *Scratch* (ou plutôt une de ses extensions appelée *mBlock*). *Scratch* est un environnement de programmation spécialement conçu pour l'éducation : très intuitif, gratuit, multiplateforme... il a de nombreuses qualités pour servir de base à l'apprentissage de la programmation³.
- Pour résumer, les différentes possibilités d'utilisation de la carte *Arduino* sont les suivantes.

3. *Blockly* constitue une excellente alternative, surtout lorsqu'il s'agit de programmer des objets physiques (robots, cartes *Arduino*...). Voir à ce sujet le projet « Robotique avec Thymio », page 300.



N. B. : dans ce schéma, *Scratch* peut être remplacé par n'importe quel autre langage graphique (*Ardublock*, *Blockly*...) permettant de piloter une carte *Arduino*.

Il existe plusieurs variantes de la carte *Arduino*, et plusieurs « *Shields* » possibles. Notre séquence est basée sur l'utilisation de la carte *Arduino Uno* et du *Shield Grove* qui possède une grande quantité de capteurs et actionneurs compatibles.

Le projet peut être mené avec d'autres variantes de carte *Arduino* ou de *Shields* ou même de langages de programmation : le déroulement sera sensiblement le même, mais les captures d'écran et photos des montages seront différentes.

Le montage du *Shield Grove* sur la carte *Arduino* est très rapide et ne présente aucune difficulté. Cette opération peut être menée par les élèves. Cependant, nous conseillons au professeur de la réaliser en amont, ne serait-ce que pour vérifier le bon fonctionnement de l'ensemble (quitte, une fois la vérification faite, à démonter le système pour laisser les élèves le remonter).

Capteurs-actionneurs

Il existe des dizaines de capteurs ou actionneurs différents, pour des prix allant de quelques euros à quelques dizaines d'euros.

Voici la liste du matériel nécessaire pour réaliser le projet tel qu'il est présenté dans les séances qui suivent⁴ :

Descriptif	capteur/ actionneur	analogique / numérique	Quantité
carte <i>Arduino Uno</i>			1 par groupe
carte <i>Shield Grove V2</i>			1 par groupe
Module <i>Grove</i> capteur sonore	Capteur	analogique	1 pour 2 groupes
Module <i>Grove</i> Détecteur de mouvement	Capteur	numérique	1 pour 2 groupes
Module <i>Grove</i> capteur de vibration	Capteur	numérique	1 pour 2 groupes
Module <i>Grove</i> capteur de lumière	Capteur	analogique	1 pour 2 groupes
Module <i>Grove</i> capteur de gaz/ fumée	Capteur	analogique	1 pour 2 groupes
Module <i>Grove</i> interrupteur magnétique (ILS) + petit aimant	Capteur	numérique	1 pour 2 groupes
Module <i>Grove</i> bouton-poussoir ou interrupteur	Capteur	numérique	1 par groupe
Module <i>Grove Buzzer</i>	Actionneur	numérique	1 par groupe
Module <i>Grove DEL</i> de diverses couleurs	Actionneur	numérique	1 verte, 1 rouge
Module <i>Grove DEL</i> blanche	Actionneur	numérique	2
Câbles <i>Grove</i> 20 cm			4
Câbles <i>Grove</i> 50 cm			4
Lot de 4 supports <i>Grove</i> 1x1			2
Lot de 4 supports <i>Grove</i> 1x2			1
Pile 9V			1 par groupe

Logiciel mBlock

Le logiciel *mBlock* est une extension de *Scratch* conçue spécifiquement pour piloter une carte *Arduino*. Ce logiciel est gratuit et disponible pour plusieurs plateformes (Windows, Mac).

Attention: dans sa version de base, à l'heure où nous écrivons ces lignes, *mBlock* fonctionne avec *Arduino*, mais pas avec ses différents « *Shields* ». Certains fabricants ou distributeurs de matériel ont développé des versions de *mBlock* optimisées pour l'utilisation couplée de carte *Arduino* et du *Shield Grove*. « Technologie Services » propose un fichier d'installation comprenant à la fois le logiciel *mBlock* et son extension compatible, ainsi qu'un guide d'installation. Nous avons utilisé *mBlock* version 3.3.1 et l'extension « technologies services » version 1.2. L'ensemble est téléchargeable ici : <http://tinyurl.com/domotique-lamap>

4. Le coût pour chaque groupe est estimé à une cinquantaine d'euros, auprès de distributeurs comme « technologie services » ou « go tronic ».

Attention : l'installation et la configuration de *mBlock* (et de sa variante adaptée au Shield Grove) prennent un peu de temps, aussi il est indispensable que ces opérations aient été réalisées par le professeur avant la séance, sur chaque ordinateur susceptible d'être utilisé par les élèves. On trouve des tutoriels (PDF ou vidéos) très bien faits sur les sites web des distributeurs.

Présentation d'Arduino

Le professeur annonce aux élèves qu'ils vont désormais utiliser des composants électroniques permettant de réaliser la maquette de leur système domotique. Il explique le fonctionnement général :

- Les capteurs et les actionneurs sont branchés sur une carte qui permet de les contrôler. Cette carte s'appelle *Arduino*.
- La carte *Arduino* peut être programmée à l'aide d'un ordinateur, en utilisant l'environnement de programmation *mBlock* (si c'est bien le choix qui a été fait!).

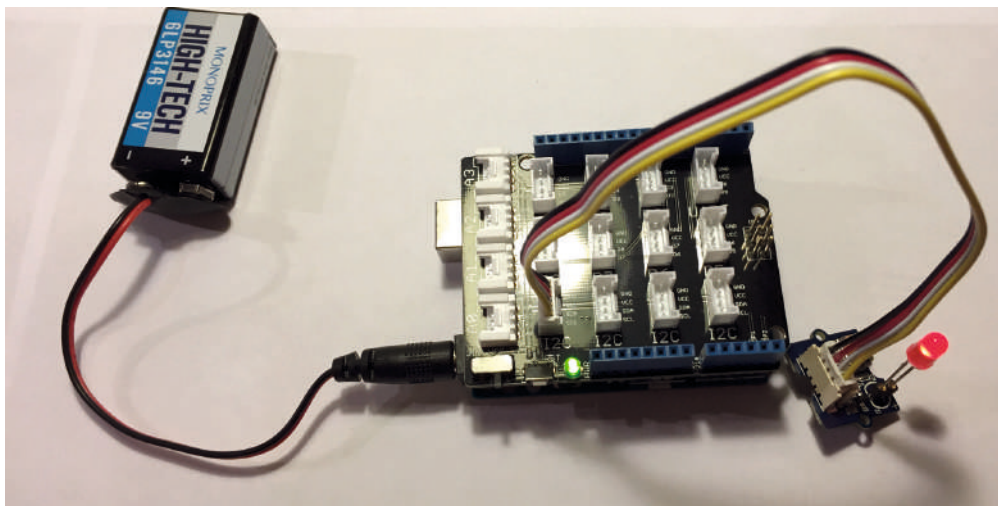
Le professeur explique qu'avant de réaliser des montages complexes, la classe va avoir besoin de se familiariser avec ce matériel, et va donc commencer par des montages très simples. Petit à petit, les montages deviendront plus complexes, tout comme les programmes qui les piloteront.

Premier montage : brancher *Arduino*, le *Shield Grove* et une DEL (par groupes)

Le professeur peut, au choix, présenter le matériel à chaque groupe et leur demander d'identifier les différents éléments, ainsi que leurs rôles, ou bien distribuer la Fiche 2.

Le premier exercice consiste simplement à trouver un branchement qui permette à la DEL rouge de s'allumer.

Les élèves peuvent tâtonner pour trouver par eux-mêmes, dès lors que le professeur les a prévenus que les pattes de branchement du *Shield Grove* sont très fragiles et à manier avec précaution.

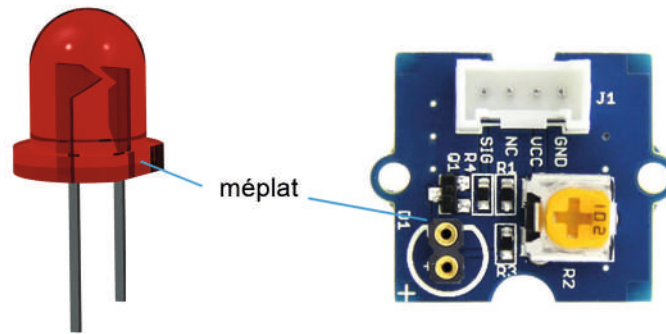


Carte *Arduino*, *Shield Grove* et DEL.

Le branchement est très simple :

1. Brancher le *Shield Grove* sur la carte *Arduino* (des détrompeurs empêchent de la brancher à l'envers).
2. Alimenter l'ensemble en branchant le câble à la pile 9V d'un côté, et à la carte *Arduino* de l'autre. Si les branchements sont corrects, 2 diodes vertes s'allument : une sur le *Shield Grove* (en haut), et une sur la carte *Arduino* (en bas).

3. Insérer la DEL sur son *socket*: la silhouette du méplat est indiquée sur l'époxy, pour éviter les erreurs de polarité.



4. Brancher le câble *Grove* sur le socket de la DEL d'un côté et sur le *Shield* de l'autre (là aussi, des détrompeurs évitent les erreurs). Tous les branchements disponibles ne correspondent pas à des sorties, et toutes les sorties ne sont pas alimentées par défaut: il faut donc tâtonner pour trouver l'emplacement correct sur le *Shield Grove* (un des emplacements marqués I2C ou UART, voire note ci-dessous).

Notes scientifiques

- Le *Shield Grove* possède 16 connecteurs 4 broches, sur lesquels on peut brancher de nombreux capteurs/actionneurs compatibles:
 - 4 connecteurs analogiques: A0, A1, A2 et A3
 - 4 connecteurs numériques (ou « digitaux »): D2, D3, D4, D5, D6, D7 et D8
 - 4 interfaces I2C (Internal *Integrated Circuit*: standard de communication entre un microprocesseur et un circuit, particulièrement utilisé dans la domotique).
 - 1 interface UART (*Universal Asynchronous Receiver Transmitter*: interface entre un microprocesseur et un port série)
- Par ailleurs, le *Shield Grove* possède une connectique autorisant le branchement d'un second *Shield Grove* par-dessus (ce qui peut être utile si l'on est limité par les 16 connecteurs du *Shield*).
- Les détails de fonctionnement de la carte *Arduino* et de son *Shield Grove* n'ont pas besoin d'être connus des élèves, tout comme la signification des différents standards de communication.

Note pédagogique

Si les élèves n'ont jamais manipulé de DEL avant ce projet, il peut être utile de consacrer quelques minutes à réaliser un branchement simple (sans carte *Arduino*), consistant à allumer une DEL à l'aide d'une pile. Ainsi, la carte *Arduino* apparaîtra moins comme une « boîte noire ».

Second montage : faire clignoter la DEL (programmer la carte *Arduino* via mBlock)

Le professeur demande aux élèves comment faire pour décider si la DEL doit s'allumer ou non. Le précédent montage ne laisse pas le choix (selon comment on l'a branchée, la DEL est soit toujours allumée, soit toujours éteinte). La solution est: programmer la carte *Arduino* pour lui dire d'allumer et d'éteindre la DEL successivement.

Branchement et découverte de Scratch/mBlock

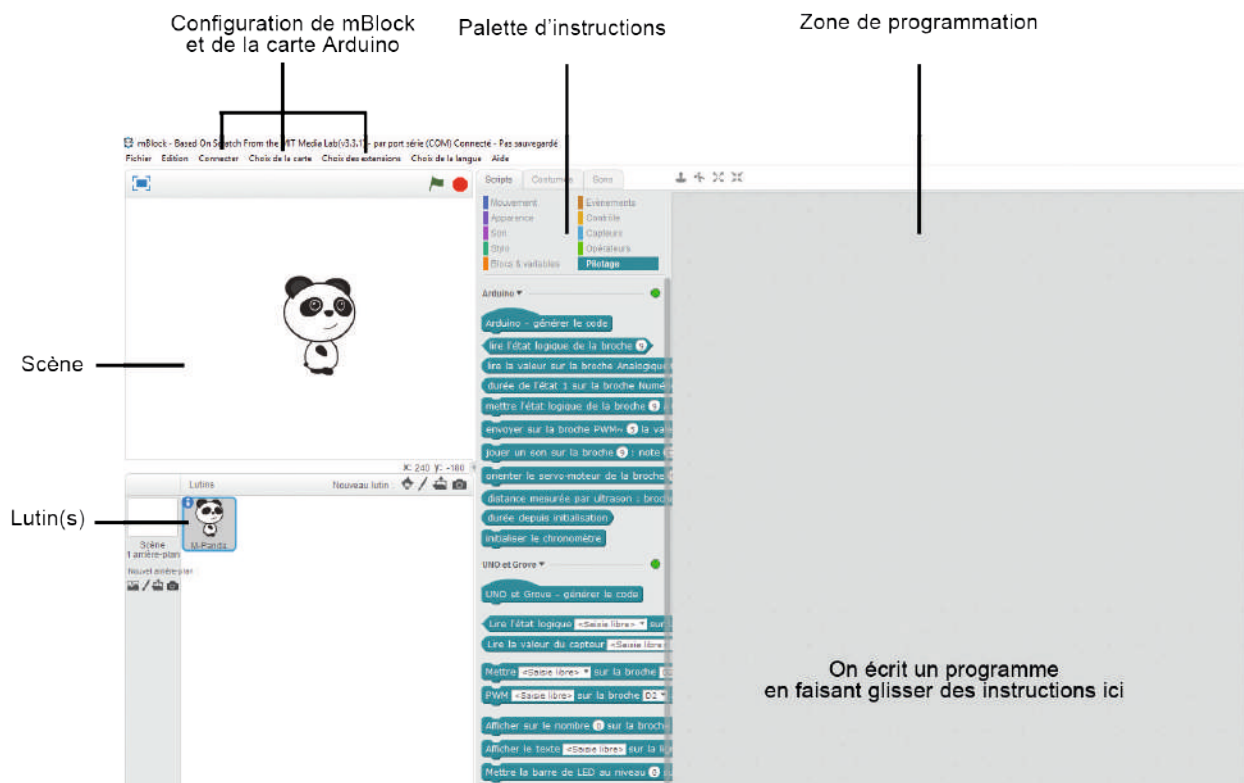
L'enseignant explique aux élèves que *Scratch* est un langage de programmation conçu spécifiquement pour apprendre à programmer, et que *mBlock* est une extension de *Scratch* conçue pour piloter des périphériques comme une carte *Arduino*.

Lorsque l'on ouvre le logiciel, il y a un lutin à l'écran (un panda). Tout comme dans *Scratch*, on peut ajouter autant de lutins qu'on le souhaite, et programmer chacun d'entre eux.

Les élèves branchent leurs cartes *Arduino* aux ordinateurs (via leurs câbles USB) et lancent le programme *mBlock*. Ils s'aident pour cela de la Fiche 3.

Note scientifique

Selon la configuration matérielle de l'ordinateur, il se peut que le port USB délivre un courant suffisant pour alimenter la carte *Arduino*, le *Shield Grove* et la DEL. Dans ce cas, l'alimentation externe 9V n'est plus nécessaire.



Interface de *mBlock*.

Le programme en exemple ici permet de faire clignoter une DEL branchée sur le port D8 du *Shield Grove*.

L'enseignant présente rapidement l'interface de *Scratch/mBlock*, qui comprend :

- Une « scène » : c'est là que se déroule le programme. Dans notre cas particulier (pilotage de la carte *Arduino*), le résultat du programme se verra davantage sur la carte elle-même que sur la scène.
- Une zone « lutins » : les lutins sont les personnages ou les objets qui seront manipulés dans le programme (ils peuvent se déplacer, changer de forme, parler, interagir avec les autres lutins...). Là encore, dans notre cas particulier, on se servira peu des lutins. En prolongement de ce projet, on peut néanmoins réaliser une modélisation du logement dans *mBlock*, auquel cas la scène et les lutins représenteront le logement, et chacun des capteurs et actionneurs.

- Un onglet «script» qui permet d'accéder à :
 - Une palette d'instructions (colonne centrale, à droite de la scène). C'est ici que l'on trouve les instructions (ou «blocs») à utiliser dans notre programme. Il y a de nombreuses instructions, regroupées par couleur (exemple: tout ce qui concerne le mouvement du lutin est dans un onglet bleu foncé, tout ce qui concerne son apparence est dans un onglet violet, etc.). L'onglet qui nous intéresse est l'onglet «pilotage», présent dans *mBlock* mais pas dans *Scratch* (cf. détails ci-dessous).
 - Une zone «programme», à droite de la palette d'instructions. C'est ici que l'on écrit le programme, tout simplement en prenant des instructions depuis la palette et en les faisant glisser dans cette zone. Le professeur réalise une rapide démonstration. Par exemple, pour demander au panda de se déplacer de 10 pas, il suffit de faire glisser l'instruction «avancer de 10» depuis la palette d'instructions vers la zone du programme. Si l'on clique ensuite sur cette inscription, on remarque que le panda avance bien de 10 pas (1 pas = 1 pixel de l'écran).



Si l'on souhaite avancer de 20 pas, il suffit de changer «10» en «20» en cliquant dans la zone dédiée. Si maintenant on souhaite que le panda avance de 20 pas, puis dise «Bonjour», il suffit de coller la nouvelle instruction à la fin du programme. L'instruction «dire bonjour» n'existe pas, mais il y a une instruction «dire Hello» dans la catégorie «apparence» de la palette d'instructions. Il suffit de prendre cette instruction puis de remplacer le texte «Hello» en «Bonjour» en cliquant sur ce texte. Écrire un programme se fait simplement en emboîtant des instructions entre elles.

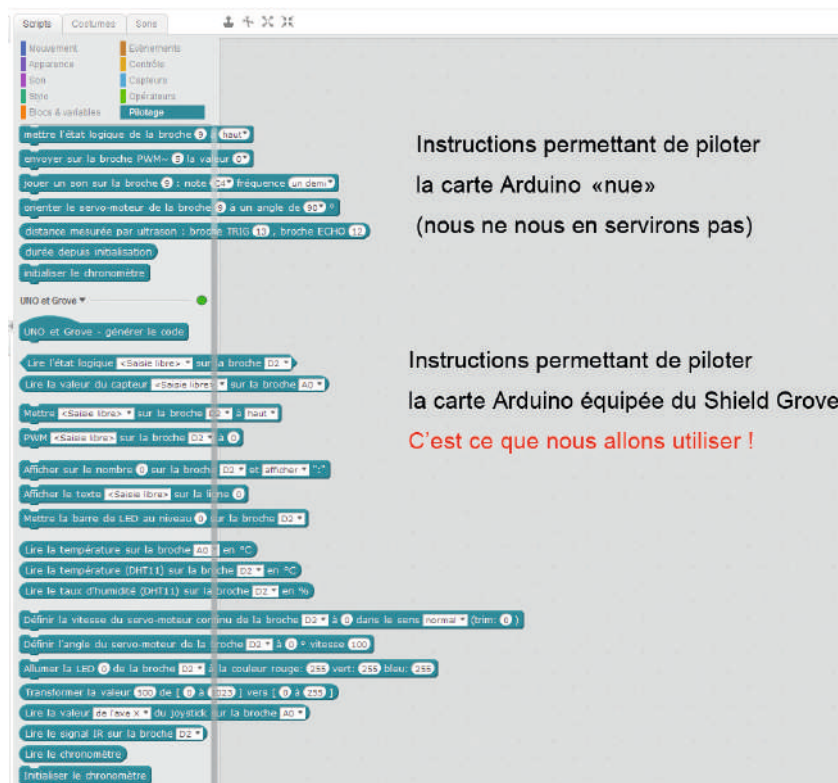


Si maintenant on veut que le panda fasse cela à chaque fois que l'on clique sur le drapeau vert (en haut à droite de la scène, le drapeau vert permet de lancer le programme), alors il faut rajouter l'instruction «Quand drapeau vert pressé» à chercher dans la catégorie «événements» des instructions. Cela donne :



Finalement, l'enseignant montre comment supprimer une instruction (ou tout un bloc d'instructions) : il suffit de faire glisser cette instruction (ou ce bloc) depuis la zone du programme vers la palette des instructions.

Ici, la palette d'instructions qui nous intéresse est la palette vert foncé appelée «pilotage». C'est elle qui contient toutes les instructions permettant de piloter la carte *Arduino* et son *Shield Grove*.



Note scientifique

Il existe deux modes permettant de piloter une carte *Arduino* via *mBlock*: le mode connecté et le mode déconnecté. Le mode connecté nécessite que la carte soit branchée physiquement à l'ordinateur, tout le temps. Le mode déconnecté permet (une fois que l'on a transféré le programme sur la carte) de débrancher la carte *Arduino* de l'ordinateur: la carte est alors autonome (penser, dans ce cas, à lui rajouter son alimentation!). Dans un premier temps, nous utiliserons le mode «connecté».

Connecter la carte Arduino

La Fiche 3 explique comment connecter la carte *Arduino* à *mBlock*.

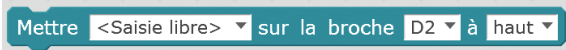
Allumer et éteindre la DEL depuis mBlock

Le professeur explique que les ports I2C et UART (utilisés précédemment) ne sont pas pilotables depuis *mBlock*. Il faut donc utiliser les ports A0...A3 ou D2...D8.

- Les ports analogiques A0...A3 ne sont utilisables qu'en entrée (on peut lire des valeurs, mais on ne peut pas agir sur le capteur via ces ports).
- Les ports numériques (ou digitaux) D2...D8 sont utilisables à la fois en entrée et en sortie. C'est sur l'un deux que l'on va brancher notre DEL, puisqu'on souhaite modifier son état.

Les élèves branchent donc leur DEL sur l'un des ports D2 à D8.

Le professeur montre le principe de la programmation *mBlock* en faisant une petite démonstration: il

fait glisser la commande  dans la zone de programme. Il remplace la valeur «D2» par celle correspondant au port sur lequel la DEL est branchée, et personnalise

le message en choisissant «DEL rouge» (ce message n'est pas interprété par le programme, il sert à faciliter la lecture du programme par l'utilisateur).

En cliquant, ensuite, sur cette instruction, on remarque que la DEL s'allume. Si on remplace, dans l'instruction, la valeur «haut» par «bas», et qu'on recliqe sur le block, alors la DEL s'éteint.

Les élèves répètent cette manipulation afin de se familiariser avec l'interface et les commandes.

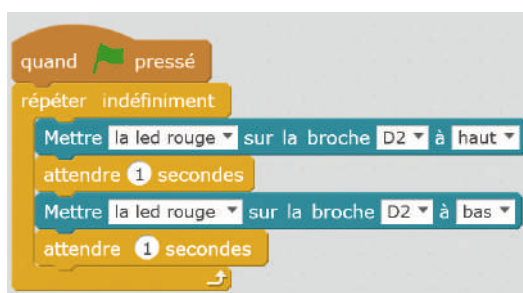
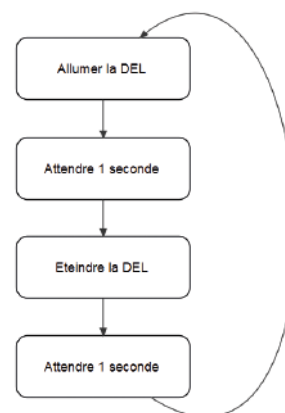
Faire clignoter la DEL depuis mBlock

Le professeur demande aux élèves comment faire pour que la DEL clignote. En cas de besoin, il peut les inciter à expliciter l'algorithme avant de chercher à le programmer (c'est, en général, une très bonne habitude!).

L'algorithme s'exprime simplement de la façon suivante :

- Répéter sans arrêt
 - Allumer la DEL
 - Attendre 1 seconde (évidemment, cette valeur est arbitraire!)
 - Eteindre la DEL
 - Attendre 1 seconde

Le professeur laisse les élèves tâtonner pour programmer cet algorithme à l'aide des instructions *mBlock*. En cas de difficulté, il peut leur indiquer que les structures de contrôles (pauses, boucles...) se trouvent dans la palette jaune.



Programme *mBlock* permettant de faire clignoter la DEL branchée sur la carte *Arduino*.

Bilan et conclusion

Un bilan s'impose car les élèves ont abordé beaucoup de notions nouvelles au cours de cette séance. La classe revient collectivement sur ce qui a été appris :

- *Arduino* est une carte électronique permettant de piloter des capteurs et des actionneurs. Cette carte a besoin d'une alimentation électrique (9 V).
- Le *Shield Grove* permet de faciliter les branchements sur la carte *Arduino*.
- La carte *Arduino* peut être programmée à l'aide d'un ordinateur, par exemple en utilisant l'environnement de programmation *mBlock*.
- *mBlock* permet de construire des programmes en assemblant des blocs d'instructions.

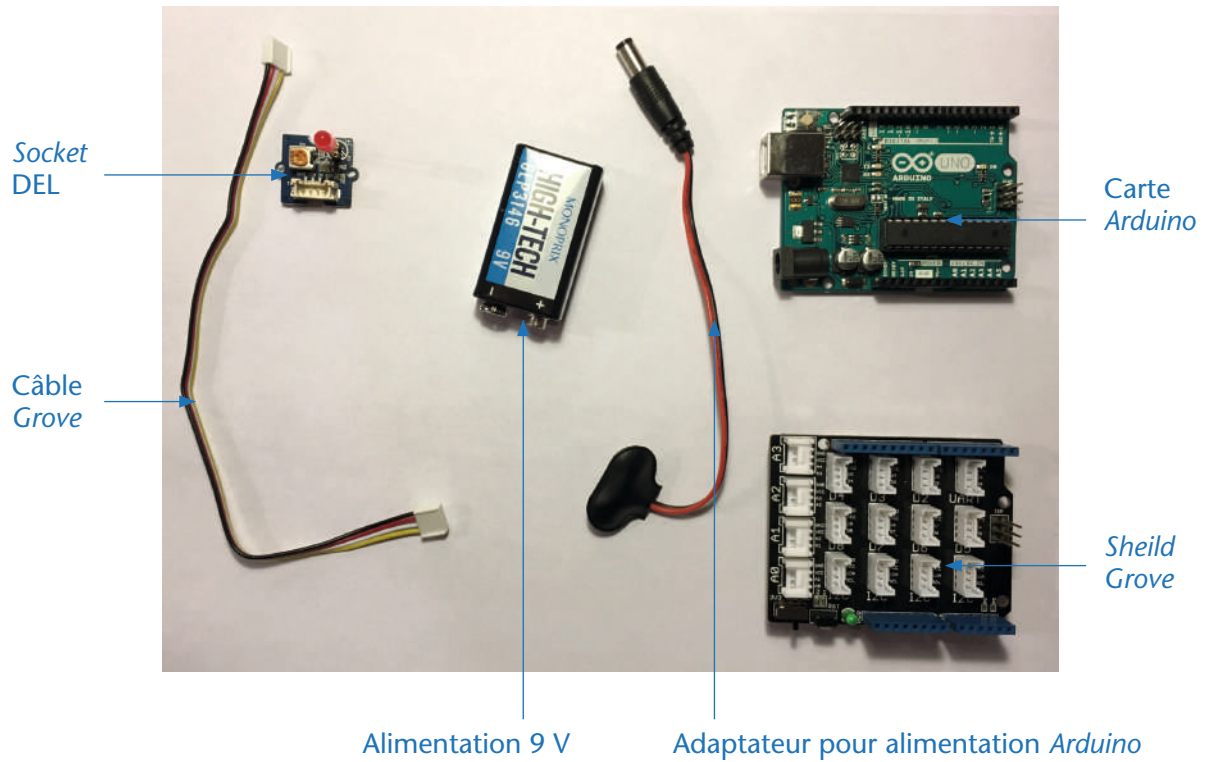
La classe synthétise également les commandes *mBlock* qui sont désormais connues de tous :

- Quand drapeau vert est pressé
- Répéter indéfiniment
- Attendre... secondes
- Mettre... sur la broche... à...

Comme lors des séances précédentes, les élèves archivent les notes et photos qu'ils ont prises dans le but de préparer la restitution finale.

FICHE 2

Montage simple à l'aide d'Arduino

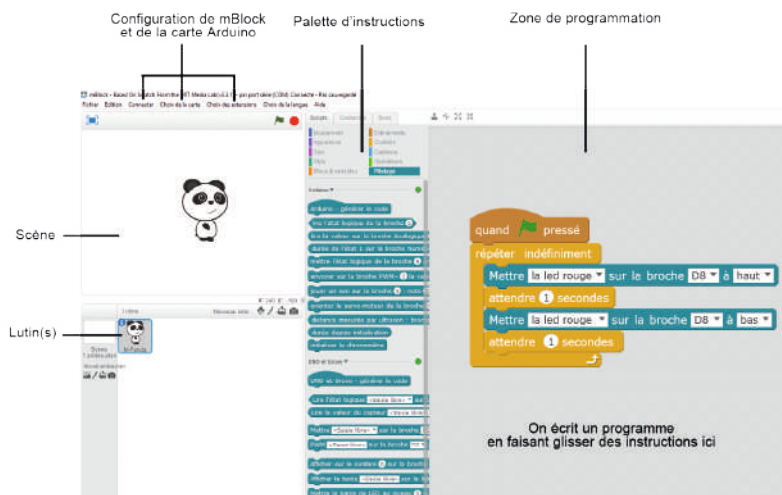


Consigne : Branche les différents éléments de façon à allumer la DEL rouge.

FICHE 3

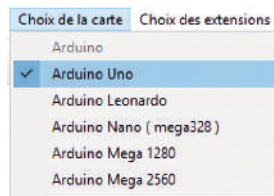
Piloter une carte *Arduino* avec *mBlock*

Le logiciel *mBlock* est une extension de *Scratch* permettant de piloter une carte *Arduino*. La programmation se fait à l'aide d'instructions qui sont des blocs que l'on assemble entre eux.

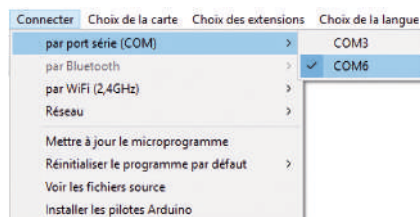
**Connecter la carte *Arduino***

Pour que *mBlock* puisse communiquer avec la carte *Arduino*, il faut :

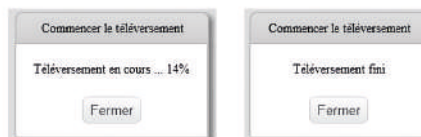
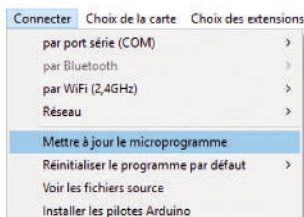
1. Cliquer sur le menu « choix de la carte » puis « *Arduino Uno* »



2. Choisir le port de communication (la valeur du port COM à choisir dépend de l'endroit où l'on a branché le câble USB)
N. B. : une fois cette étape réalisée, le mot « connecté » apparaît dans la barre de titre de la fenêtre *mBlock*.



3. Mettre à jour le microprogramme (si le téléversement n'avance pas, le choix du port COM n'était sans doute pas le bon).



Allumer une DEL

- Brancher une DEL sur un port numérique du *Shield Grove* (par exemple, le port D2).
- Faire glisser l'instruction suivante dans la zone de programmation, puis cliquer dessus pour l'exécuter.



- Personnaliser le texte « saisie libre » pour améliorer la lisibilité du programme.
- Brancher la DEL sur un autre port (par exemple, D7), et mettre à jour le programme.



Séance 4 – Étude de quelques capteurs : analogiques ou numériques ?

Discipline dominante	Technologie
Résumé	Les élèves étudient différents capteurs à l'aide de montages très simples : modalités de branchement, d'interrogation et de contrôle, valeurs renvoyées par ces capteurs. Ils distinguent alors les signaux analogiques et numériques. Enfin, ils apprennent à réaliser un montage plus complet permettant au microcontrôleur de piloter un actionneur à partir des informations fournies par un capteur.
Notions (cf. scénario conceptuel, page 195)	Algorithmes : <ul style="list-style-type: none">• Un test permet de choisir quelle action effectuer si une condition est vérifiée ou non. Information : <ul style="list-style-type: none">• Un signal est composé des différentes valeurs d'une information au cours du temps.• Certains signaux, dits « analogiques », peuvent prendre une infinité de valeurs différentes : ils varient de façon continue.• D'autres signaux, dits « numériques », ne prennent que quelques valeurs différentes (souvent 2) : ils varient de façon discontinue.
Type d'activité	Séance branchée (robotique)
Matériel	Pour chaque groupe : <ul style="list-style-type: none">• Un ordinateur sur lequel a été installé le logiciel <i>mBlock</i>• Une carte <i>Arduino</i> et son <i>Shield Grove</i> + alimentation et connectique• 2 modules <i>Grove</i> (capteurs)• 1 module <i>Grove</i> (actionneur : DEL ou buzzer)• Fiche 4, page 222• Un appareil photo numérique

Rappel de la séance précédente

Le professeur demande aux élèves de refaire le montage (et le programme) réalisé à la séance précédente, consistant à faire clignoter une DEL. Cette reprise leur permet de consolider les différentes notions, dont la maîtrise est nécessaire pour la suite.

Situation déclenchante

Le professeur rappelle l'objectif du projet (sécuriser une maison), et demande aux élèves les fonctions qui avaient été évoquées au début de cette séquence. L'objectif est de lister les différents capteurs et actionneurs dont la classe va avoir besoin.

Chaque classe possède son propre projet (cf. cahiers des charges établis précédemment), et donc sa propre liste de capteurs/actionneurs. À titre indicatif, voici quelques propositions de modules compatibles *Arduino/Shield Grove* :

- Capteurs
- mouvement

- son
- lumière
- vibration
- gaz/fumée
- ouverture (interrupteur magnétique ILS)
- bouton-poussoir ou interrupteur
- Actionneurs :
 - DEL verte, rouge, blanche
 - buzzer

On peut aussi ajouter, par exemple, une horloge (permettant de programmer l'alarme ou la simulation de présence), un haut-parleur en remplacement du buzzer (ce qui permet d'émettre des sons de fréquences variées), voire une caméra pour de la vidéosurveillance (attention, la caméra *Grove* nécessite, en plus du *Shield Grove*, un *Shield carte SD* pour fonctionner), etc.

Le professeur indique que chaque groupe va étudier le fonctionnement de quelques capteurs et actionneurs. Il explique que certains capteurs ne sont capables de renvoyer que 2 valeurs distinctes (par exemple, 0 et 1), tandis que d'autres sont capables de renvoyer une grande quantité de valeurs comprises entre 2 nombres (par exemple, entre 1 et 100). Il explique qu'on appelle les premiers des capteurs « numériques » (ou « digitaux » pour reprendre un anglicisme très répandu) et les seconds des capteurs « analogiques » ; il fait le rapprochement avec les connexions disponibles sur le *Shield Grove* :

- les connexions A0...A3 sont pour les capteurs analogiques (A comme Analogique)
- les connexions D2...D8 sont pour les capteurs (ou actionneurs) numériques/digitaux (D comme Digital)

Note scientifique

un signal analogique varie de façon continue: il y a donc une infinité de valeurs possibles (par exemple, tous les nombres réels entre 1 et 100). Cependant, ce signal, pour être traité par un ordinateur, doit être numérisé. Cette numérisation impose que l'on discrétise les valeurs possibles: elles ne sont plus en nombre infini. Pour plus d'information sur la numérisation, voir l'éclairage scientifique page 44 ainsi que le projet EPI « Conception et programmation d'un synthétiseur », page 242.

Recherche (par groupe) : analogique ou numérique ?

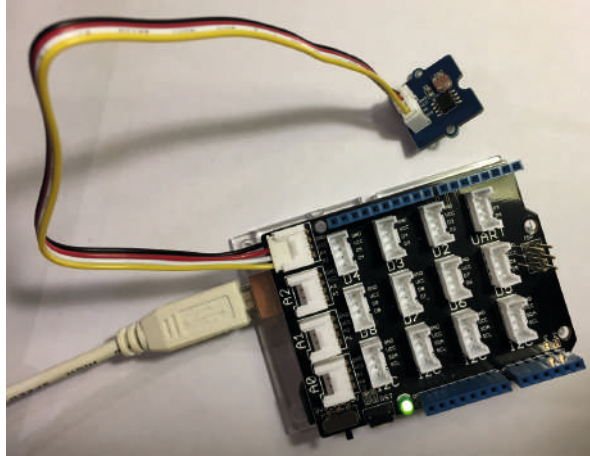
Chaque groupe reçoit 2 capteurs différents (en plus de la carte *Arduino* équipée de son *Shield*, de l'alimentation et de la connectique nécessaire) parmi les capteurs utilisés dans le projet (mouvement, son, lumière, vibration, gaz/fumée, magnétique, bouton-poussoir/interrupteur...).

Les élèves doivent remplir la Fiche 4, et pour cela tester différentes façons de brancher leurs capteurs sur le *Shield* (soit sur une connexion analogique, soit sur une connexion numérique). Ils doivent par ailleurs expliquer ce que fait ce capteur, et indiquer les différentes valeurs qu'il retourne. Pour cela, il est nécessaire d'écrire un programme *mBlock* permettant de visualiser ces valeurs.

En fonction de l'aisance des élèves, le professeur pourra les laisser en complète autonomie pour la phase de programmation, ou leur indiquer simplement les instructions nécessaires (mais en les mettant dans le désordre), ou encore faire une démonstration collective pour un capteur (les élèves devant, ensuite, s'inspirer de cette méthode pour leurs propres investigations).

Exemple 1: le capteur de lumière (analogique)

Ce capteur se branche sur une entrée analogique. Par exemple A03 :



Branchement du capteur de lumière

L'instruction permettant d'afficher la valeur retournée par le capteur de lumière est, tout simplement :

Lire la valeur du capteur lumière sur la broche A3

En cliquant sur cette instruction, on affiche la valeur retournée par le capteur.

En cliquant sur cette instruction, on affiche la valeur retournée par le capteur.

Pour faire afficher cette valeur de façon continue, il y a deux possibilités :

- faire « dire » cette valeur par le lutin (commande « dire » dans l'onglet « apparence ») : la valeur est affichée dans une bulle ;
- créer une variable, lui affecter la valeur qui nous intéresse, et faire afficher cette variable à l'écran.

La première méthode, beaucoup plus simple, suffit amplement pour cette tâche. Si l'on souhaite que le programme affiche cette valeur en continu, le programme *mBlock* devient :



Programme simple permettant d'afficher la valeur retournée par le capteur de lumière.

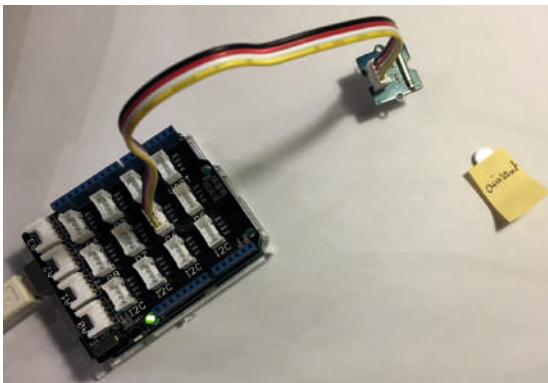
Si l'on place le capteur au creux de ses mains ou en pleine lumière, on remarque qu'il renvoie des valeurs allant de 0 à environ 800 (variable selon l'intensité de la lumière).

Notes scientifiques

- Tous les capteurs analogiques pour *Arduino* renvoient des valeurs comprises entre 0 et 1023 (il y a 2^{10} valeurs différentes, car il s'agit d'un encodage sur 10 bits).
- Certains capteurs (qu'ils soient analogiques ou numériques) possèdent une molette permettant de régler leur sensibilité via un potentiomètre. Cette molette est en général réglable à l'aide d'un petit tournevis cruciforme.

Exemple 2 : l'interrupteur magnétique ILS (numérique)

Ce capteur est un capteur numérique, qui ne renvoie que 2 valeurs différentes : 1, lorsqu'il détecte un aimant à proximité, ou 0 s'il ne détecte rien.



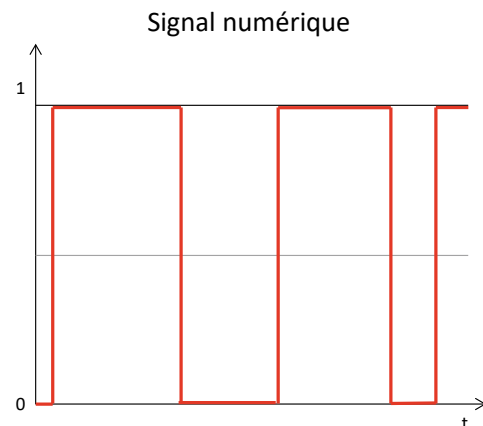
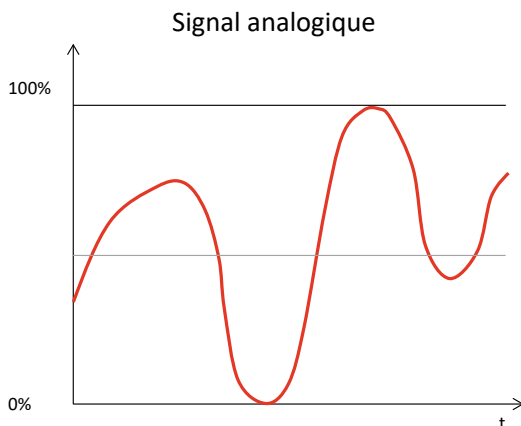
Branchement et programme permettant de tester l'interrupteur magnétique ILS.
Si on rapproche le petit aimant du capteur, la valeur renvoyée par le capteur passe de 0 à 1.

Les autres capteurs donnent lieu à des travaux similaires.

Mise en commun (collectivement)

Le professeur anime la mise en commun au cours de laquelle chaque porte-parole présente les résultats de son groupe. Les différents capteurs sont comparés, et on vérifie que certains renvoient bien une plage de valeurs continues (les capteurs analogiques) tandis que d'autres ne renvoient que deux valeurs possibles (les capteurs numériques).

Le professeur introduit la notion de signal, qui est la variation dans le temps d'une certaine information (par exemple, température, luminosité ambiante...). Selon la manière dont un signal varie, on parle de signal analogique ou de signal numérique.

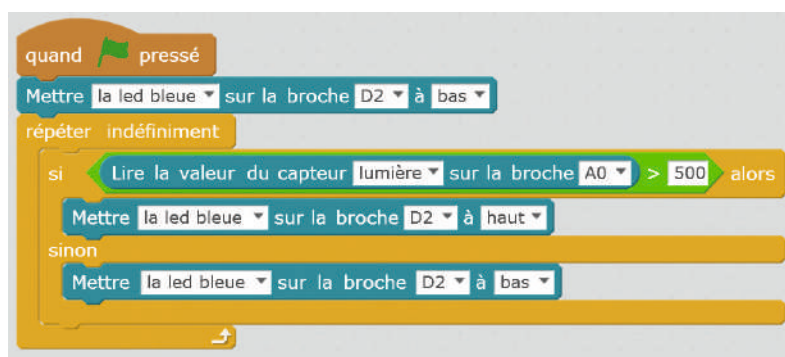


Recherche (par groupes) : commander un actionneur à l'aide d'un capteur

Les groupes reçoivent désormais un actionneur, en plus de leur(s) capteur(s), avec la consigne de faire marcher l'actionneur uniquement dans le cas où la valeur renvoyée par le capteur dépasse un certain seuil. Par exemple :

- Le buzzer ne doit bipper que si la luminosité ambiante dépasse la valeur seuil de 500 (sur une échelle de 0 à 1023).
- La DEL ne doit s'allumer que si le bouton-poussoir est enfoncé (ce bouton-poussoir peut renvoyer 2 valeurs différentes: VRAI s'il est enfoncé, FAUX sinon), etc.

Le montage fait maintenant intervenir deux modules *Grove* branchés en même temps : le capteur et l'actionneur.



À chaque répétition de la boucle, on teste la valeur de la luminosité, et on allume la DEL si cette valeur dépasse la valeur seuil (on éteint la DEL dans le cas contraire). L'initialisation en début de programme n'est pas indispensable, mais c'est une bonne habitude à prendre.

A chaque répétition de la boucle, on teste la valeur de la luminosité, et on allume la DEL si cette valeur dépasse la valeur seuil (on éteint la DEL dans le cas contraire). L'initialisation en début de programme n'est pas indispensable, mais c'est une bonne habitude à prendre.

De la même façon que précédemment, le professeur peut laisser les élèves définir l'algorithme, puis le programmer, en autonomie complète (si ceux-ci ont déjà une petite expérience en *Scratch*, cela ne pose pas de difficulté), ou les guider, par exemple en leur indiquant les instructions nécessaires (mais sans les assembler et surtout sans les mettre dans l'ordre).

Bilan et conclusion

La classe dresse collectivement un bilan des notions abordées au cours de cette séance :

- Il existe différents types d'éléments pouvant être branchés sur la carte *Arduino* : des capteurs (qui renvoient des signaux) et des actionneurs (qui agissent sur l'environnement externe, par exemple en émettant un son, une lumière...)
- Un signal est composé des différentes valeurs d'une information au cours du temps. Certains signaux, dits analogiques, peuvent prendre une infinité de valeurs différentes : ils varient de façon continue. D'autres signaux, dits numériques, ne prennent que quelques valeurs différentes (souvent 2) : ils varient de façon discontinue.

Les élèves notent également les nouvelles commandes *mBlock* qu'ils ont appris à utiliser :

- Dire...
- Lire l'état logique... sur la broche...
- Lire la valeur du capteur... sur la broche...
- Si... alors... sinon...

Comme lors des séances précédentes, les élèves archivent les notes et photos qu'ils ont prises dans le but de préparer la restitution finale.

FICHE 4
Étude de capteurs *Arduino / Grove*

Consigne : Pour chaque actionneur, trouve la façon dont il se branche sur le « *Shield Grove* » de la carte *Arduino*, écris un programme simple permettant de vérifier son fonctionnement, et précise s'il s'agit d'un capteur analogique ou numérique.

Capteur	Plage de valeurs retournées par ce capteur	Analogique ou numérique ?



Séance 5 – Comportements complexes : opérateurs logiques

Discipline dominante	Technologie
Résumé	Les élèves apprennent à manipuler des opérateurs logiques, en particulier ET et OU, de façon à pouvoir programmer des comportements plus complexes, en lien avec les cahiers des charges produits en début de projet. Ils réalisent les montages et les programmes correspondant. NB: cette séance peut être dédoublée si les élèves ont des difficultés à manipuler les opérateurs logiques (cf. activité de prolongement conseillée).
Notions (cf. scénario conceptuel, page 195)	<p>Machines :</p> <ul style="list-style-type: none"> • En combinant plusieurs instructions simples on peut effectuer une tâche complexe. <p>Algorithmes :</p> <ul style="list-style-type: none"> • Une condition est une expression qui est soit vraie, soit fausse (on peut la représenter, parfois, sous la forme 1, pour vrai, et 0, pour faux). • On peut utiliser des opérateurs logiques (ET, OU, NON) pour fabriquer des expressions logiques.
Type d'activité	Séance branchée (robotique)
Matériel	<p>Pour chaque groupe :</p> <ul style="list-style-type: none"> • Un ordinateur sur lequel a été installé le logiciel <i>mBlock</i> • Une carte <i>Arduino</i> et son <i>Shield Grove</i> + alimentation et connectique • 2 capteurs <i>Grove</i> • 3 actionneurs <i>Grove</i> (DEL1, DEL2, buzzer) • Un appareil photo numérique <p>Pour chaque élève (prolongement conseillé) :</p> <ul style="list-style-type: none"> • Fiche 5, page 228

Situation déclenchante

La classe revient collectivement sur ce qui a été appris lors des dernières séances: comment brancher la carte *Arduino*, comment piloter un actionneur à l'aide d'un signal envoyé par un capteur à l'aide du logiciel *mBlock*, etc.

Le professeur rappelle que, jusqu'à présent, les programmes qui ont été réalisés sont des programmes élémentaires, alors que les fonctionnalités que l'on souhaite développer sont plus complexes.

Par exemple, pour le bouton « appel d'urgence »: cet appel doit être déclenché, non pas si on appuie sur 1 bouton, mais si on appuie sur l'un OU l'autre des 2 boutons présents dans la maison (un dans la chambre, un dans la salle de bains, par exemple).

Recherche (par groupes) : programmer le bouton d'alarme

Les élèves cherchent comment réaliser le branchement (il faut 2 capteurs et 1 actionneur), et comment programmer la fonctionnalité dans *mBlock*.

Au besoin, le professeur leur indique que la solution se trouve dans l'onglet « opérateurs » (vert).



Ici, le buzzer se déclenche si l'on appuie sur le bouton-poussoir OU si l'on appuie sur la touche tactile. En cliquant sur chaque instruction « lire l'état logique... », on peut vérifier que les valeurs prises sont 0/1 ou *False/True*.

Note pédagogique

Il existe des cartes compatibles *Arduino* permettant d'envoyer des SMS, ou des modules *Grove* qui permettent d'utiliser des fonctionnalités réseau (Wifi, bluetooth...) mais, pour la simplicité du projet et pour des raisons économiques, nous considérons qu'un Buzzer ou une DEL suffisent pour simuler l'appel d'urgence.

Mise en commun (collectivement)

Le professeur organise une mise en commun et s'assure que chacun a bien compris comment combiner différentes valeurs pour fabriquer une expression logique.

Un état logique peut être VRAI ou FAUX. Un opérateur logique (OU, ET ou NON) permet de combiner ces différents états logiques pour fabriquer une expression plus ou moins complexe, qui possède elle-même une valeur VRAI ou FAUX.

- Le résultat de (a) OU (b) est VRAI si (a) est VRAI ou si (b) est VRAI (ou les deux à la fois : ce OU n'est pas exclusif).
- Le résultat de (a) ET (b) est VRAI si et seulement si (a) et (b) sont VRAIS tous les deux à la fois.
- Le résultat de NON (a) est VRAI si et seulement si (a) est FAUX.

Notes pédagogiques

- Il n'est pas nécessaire (et pas dans les programmes...) de faire un cours précis sur l'algèbre de Boole ou de dessiner les tables de valeurs logiques pour les différents opérateurs. Cependant, une compréhension générale (comme ici), sera très utile pour tous les travaux en algorithmique ou en programmation.
- En cas de difficultés, nous conseillons très fortement de faire une pause dans le projet et de réaliser, collectivement, l'activité de prolongement proposée en fin de cette séance. Dans le cas contraire (si les élèves n'ont pas de difficulté particulière), l'activité pourra davantage être proposée comme exercice de consolidation, à faire à la maison.
- En *Scratch/mBlock*, les états logiques ont tous la même apparence : ce sont des blocs hexagonaux. On remarque d'ailleurs que l'expression `4 < 5` est une expression logique. On peut faire afficher sa valeur (qui est : VRAI) en cliquant dessus. Les nombres « 4 » et « 5 » peuvent être remplacés par n'importe quelle valeur : variable, signal envoyé par un capteur, etc.

Afin de vérifier la bonne compréhension de cette nouvelle notion, le professeur demande aux élèves de programmer une nouvelle fonctionnalité du même type. Par exemple, pour assurer la sécurité des personnes dans la maison, on peut souhaiter faire allumer la lumière si un mouvement est détecté ET s'il fait nuit.

Recherche (par groupes)

Les élèves cherchent comment programmer cette fonctionnalité. Il faut 2 capteurs (mouvement + luminosité) et un actionneur (DEL).



Dans cet exemple, la lumière ne s'allume que si 2 conditions sont respectées simultanément: une présence est détectée ET il fait nuit (la luminosité est inférieure à un certain seuil, que l'on détermine en fonction du capteur et du lieu).

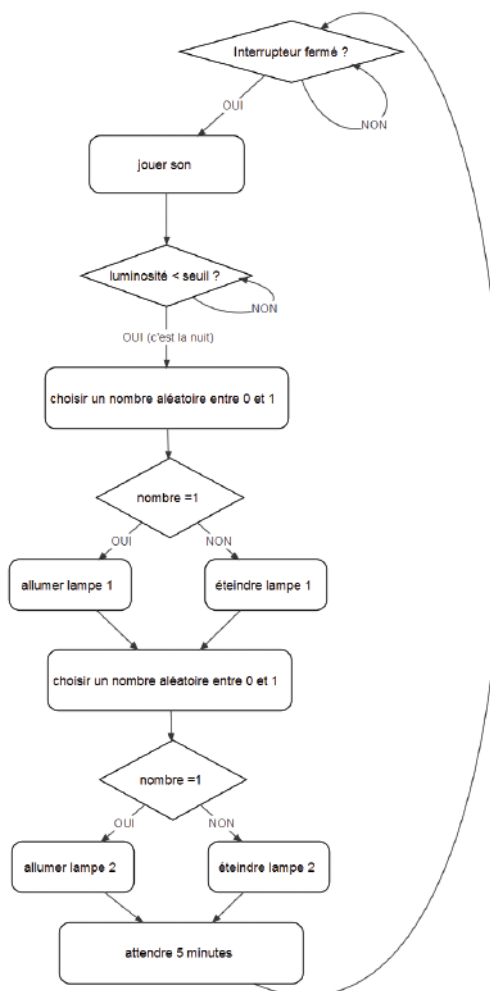
Recherche (collectivement)

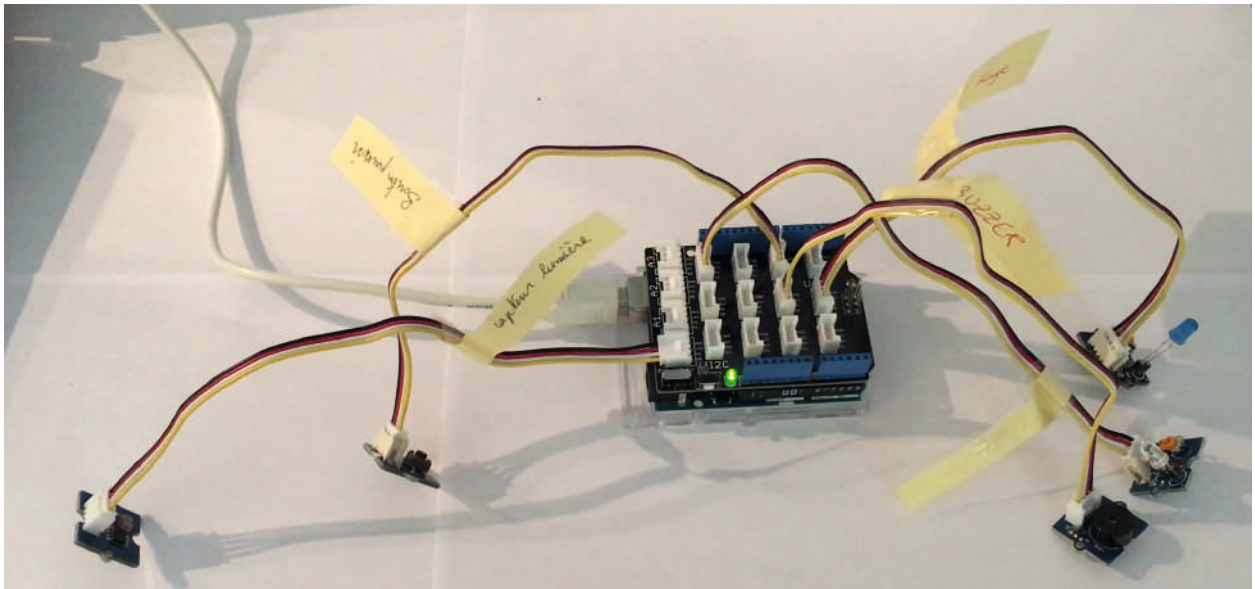
Après avoir organisé la mise en commun et s'être assuré que l'étape précédente ne pose plus de difficulté majeure, le professeur propose de résoudre collectivement une tâche plus complexe: programmer la simulation de présence. Bien sûr, en fonction de l'aisance des élèves (et du temps disponible), on peut choisir de les faire travailler en autonomie.

Cette fonction étant plus complexe, il importe de se remettre en mémoire l'algorithme précis (cf. Séance 2):

Notes pédagogiques

- Encore une fois, on adapte notre projet au matériel disponible. Il n'est pas nécessaire d'acheter un coûteux module permettant de jouer des sons MP3... pour l'exercice, un buzzer peut suffire. On peut différencier la musique d'ambiance et le son de l'alarme en utilisant à la fois un buzzer et un haut-parleur (qui permet de moduler la fréquence du son), si l'on dispose de ces 2 actionneurs.
- Comme on le verra ci-dessous, l'algorithme est relativement complexe, avec plusieurs niveaux d'imbrication qui s'ajoutent à une nouveauté: les tirages aléatoires. Ne pas hésiter à le simplifier au besoin (voir ci-dessous pour un exemple).





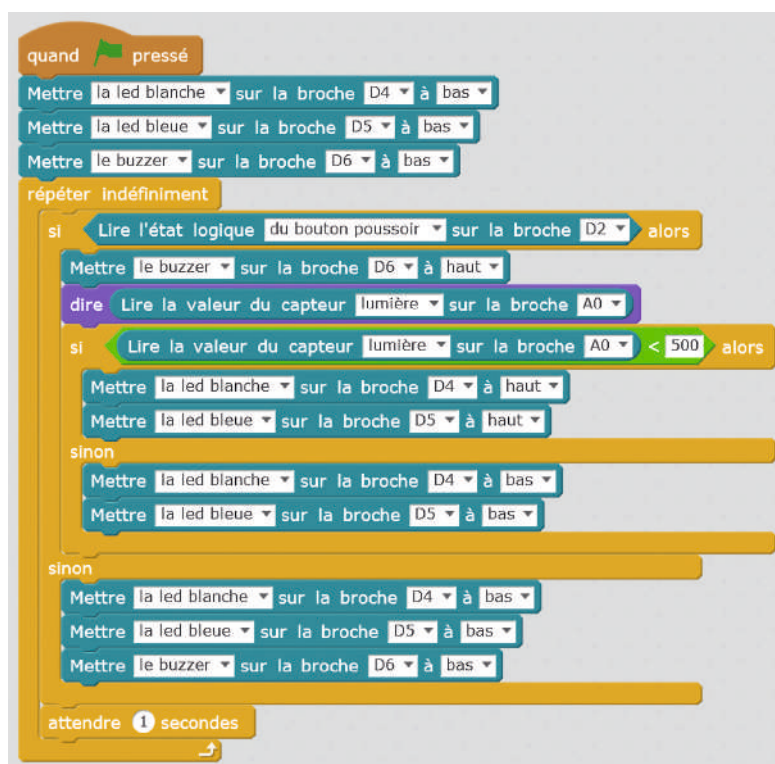
Le montage commence à être complexe : 2 capteurs (interrupteur / luminosité) et 3 actionneurs (buzzer, DEL 1, DEL2). Des étiquettes indiquent qui fait quoi (on peut améliorer ces étiquettes en prenant un code couleur pour les capteurs et un autre pour les actionneurs). Ici, les capteurs sont à gauche, et les actionneurs à droite.
 N.B. : une vidéo de ce montage en fonctionnement est disponible sur le site Web du projet.

```

quand [drapeau] pressé
  Mettre la led blanche sur la broche D4 à bas
  Mettre la led bleue sur la broche D5 à bas
  Mettre le buzzer sur la broche D6 à bas
  répéter indéfiniment
    si Lire l'état logique du bouton poussoir sur la broche D2 alors
      Mettre le buzzer sur la broche D6 à haut
      dire Lire la valeur du capteur lumière sur la broche A0
      si Lire la valeur du capteur lumière sur la broche A0 < 500 alors
        si nombre aléatoire entre 0 et 1 = 1 alors
          Mettre la led blanche sur la broche D4 à haut
        sinon
          Mettre la led blanche sur la broche D4 à bas
        si nombre aléatoire entre 0 et 1 = 1 alors
          Mettre la led bleue sur la broche D5 à haut
        sinon
          Mettre la led bleue sur la broche D5 à bas
      sinon
        Mettre la led blanche sur la broche D4 à bas
        Mettre la led bleue sur la broche D5 à bas
    sinon
      Mettre la led blanche sur la broche D4 à bas
      Mettre la led bleue sur la broche D5 à bas
      Mettre le buzzer sur la broche D6 à bas
  attendre 1 secondes
  
```

Le programme, lui aussi, est assez complexe, avec une boucle et 3 instructions conditionnelles (si...alors...sinon...) imbriquées les unes dans les autres. Noter que nous avons introduit une pause pour éviter le clignotement trop rapide, et que nous avons fait afficher la luminosité (sur la scène Scratch) afin de vérifier que le programme fait bien ce qu'il est censé faire lorsque la luminosité dépasse le seuil indiqué.

Ne pas hésiter, si cela est trop difficile pour les élèves, à simplifier l’algorithme, et donc le programme. Si on se contente d’allumer les 2 DEL en même temps, sans tirage aléatoire, le programme devient :



Version simplifiée du programme précédent (pas de tirage aléatoire).

Bilan et conclusion

La classe dresse collectivement le bilan de ce qui a été appris :

- Une condition est une expression qui est soit vraie, soit fausse (on peut la représenter, parfois, sous la forme 1, pour vrai, et 0, pour faux) ;
- On peut utiliser des opérateurs logiques (ET, OU, NON) pour fabriquer des expressions logiques.

De même que précédemment, les élèves notent les nouvelles commandes qu’ils ont appris à manipuler dans *mBlock/Scratch* :

- Nombre aléatoire entre... et...
- ... ET...
- ... OU...
- NON...

Enfin, ils archivent leurs notes et photos dans l’optique de la présentation finale du projet.

Prolongement sur les opérateurs logiques

Nous conseillons fortement une petite activité de prolongement destinée à faciliter la compréhension des opérateurs logiques. La Fiche 5 peut être étudiée en classe (prévoir une séance dédiée) ou sous la forme de devoirs à la maison (qui devront être corrigés collectivement).

FICHE 5 Manipuler quelques opérateurs logiques

Exercice 1: Les deux expressions ci-dessous sont-elles équivalentes ou différentes?

```

Lire l'état logique du bouton poussoir sur la broche D2
Lire l'état logique du bouton poussoir sur la broche D2 = VRAI
    
```

Exercice 2: Indique, pour chaque cas, si la DEL est allumée ou éteinte, ou si on ne peut pas le savoir.

```

si 3 > 10 alors
  Mettre la led rouge sur la broche D2 à haut
sinon
  Mettre la led rouge sur la broche D2 à bas

si 3 > 10 alors
  Mettre la led rouge sur la broche D2 à haut

Mettre la led rouge sur la broche D2 à bas
si 3 > 10 alors
  Mettre la led rouge sur la broche D2 à haut
    
```

Exercice 3: Voici un programme destiné à contrôler une lumière dans un couloir. Dire, dans chaque cas, si la DEL va s'allumer ou non.

```

répéter indéfiniment
  si Lire l'état logique du détecteur de présence sur la broche D2 et Lire la valeur du capteur lumière sur la broche A0 < 500 alors
    Mettre la led blanche sur la broche D2 à haut
  sinon
    Mettre la led blanche sur la broche D2 à bas
    
```

- | | | |
|---------|--|-------------------|
| Cas 1 : | <ul style="list-style-type: none"> • Une personne est dans le couloir • Il fait sombre | La DEL est: |
| Cas 2 : | <ul style="list-style-type: none"> • Une personne est dans le couloir • Il fait jour | La DEL est: |
| Cas 3 : | <ul style="list-style-type: none"> • Personne n'est dans le couloir • Il fait sombre | La DEL est: |

Exercice 4: même chose, mais avec le programme suivant

```

répéter indéfiniment
  si Lire l'état logique du détecteur de présence sur la broche D2 ou Lire la valeur du capteur lumière sur la broche A0 < 500 alors
    Mettre la led blanche sur la broche D2 à haut
  sinon
    Mettre la led blanche sur la broche D2 à bas
    
```

Exercice 5: Parmi les 2 précédents programmes, lequel est le plus pertinent? Pourquoi?



Séance 6 – Fabrication de la maquette complète

Discipline dominante	Technologie
Résumé	Les élèves fabriquent ou finalisent leurs maquettes incluant les éléments physiques de la maison et les solutions domotiques (carte <i>Arduino</i> , capteurs, actionneurs, câbles). Pour permettre à cette maquette de fonctionner indépendamment de l'ordinateur, et ainsi réaliser un vrai système embarqué, ils apprennent à utiliser le mode « autonome » d' <i>Arduino</i> .
Notions (cf. scénario conceptuel, page 195)	Machines : • Un système embarqué comprend des composants électroniques et informatiques destinés à exécuter une tâche précise de façon autonome.
Type d'activité	Séance branchée (robotique)
Matériel	Pour chaque groupe : • Un ordinateur sur lequel a été installé le logiciel <i>mBlock</i> • Une carte <i>Arduino</i> et son <i>Shield Grove</i> + alimentation et connectique • Tous les capteurs et actionneurs disponibles • Un appareil photo numérique • Carton plume ou maquette de maison déjà montée

Note pédagogique

L'objectif de ce projet est l'apprentissage de la robotique et de la programmation. Nous ne décrivons donc pas la fabrication de la maquette physique de la maison à l'aide de carton plume ou de PVC. L'enseignant peut, s'il le souhaite, faire fabriquer cette maquette par les élèves, ou bien leur proposer une maquette déjà prête afin de passer plus de temps sur les branchements et les programmes.

Situation déclenchante

La classe fait le bilan de ce qu'elle a appris à faire au cours des séances précédentes, et fait le point sur les fonctions domotiques qui sont déjà mises en œuvre, et celles qui restent à implémenter (qu'il s'agisse des branchements ou des algorithmes et programmes).

Le professeur explique que, désormais, les groupes vont tout intégrer de manière à fabriquer la maquette finale, intégrant toutes les fonctions (ou le plus possible!).

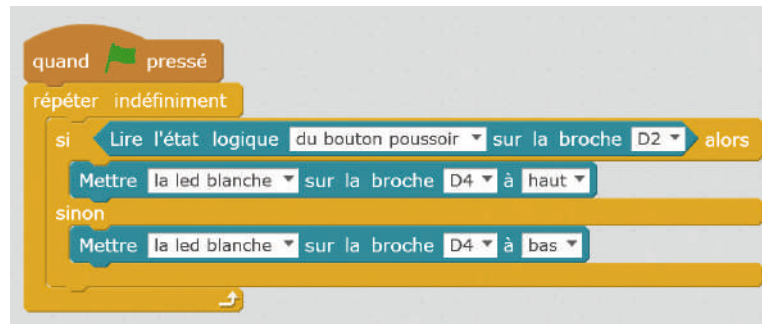
Afin que la maquette ne soit pas liée à l'ordinateur par un câble USB, il faudrait que l'on puisse débrancher la carte *Arduino* de l'ordinateur et qu'elle puisse fonctionner en autonomie.

Le professeur explique que c'est possible : la carte *Arduino* peut fonctionner en mode « connecté » (mode utilisé jusqu'à présent) et en mode « autonome » (mode que les élèves vont apprendre à utiliser maintenant).

Recherche (collectivement) : carte *Arduino* en mode autonome

Chaque groupe branche sa carte *Arduino* munie d'un capteur simple (type bouton-poussoir ou interrupteur) et d'un actionneur (DEL).

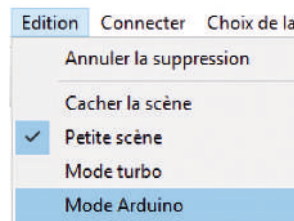
Les élèves réalisent le programme permettant d'allumer la DEL lorsque l'interrupteur est fermé (cette activité, à ce stade, ne présente plus aucune difficulté).



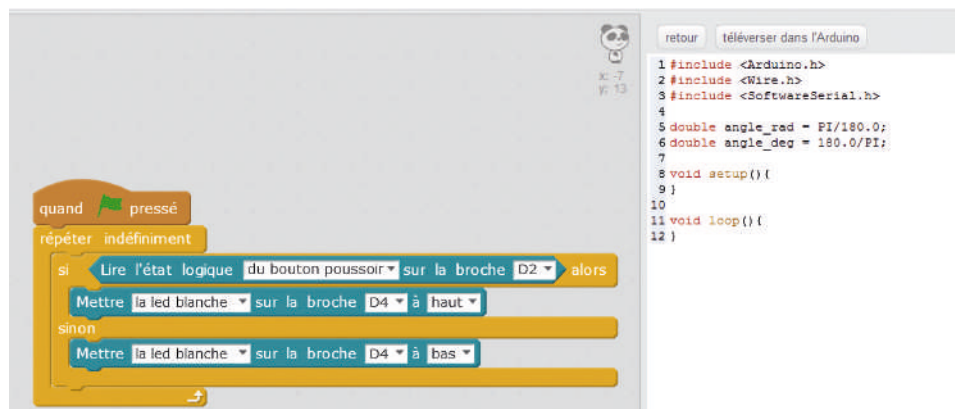
Programme permettant d'allumer la DEL à l'aide du bouton poussoir, en mode connecté.

L'enseignant explique que pour passer en mode « autonome », il faut :

1. Cliquer sur « Edition » puis « Mode Arduino »

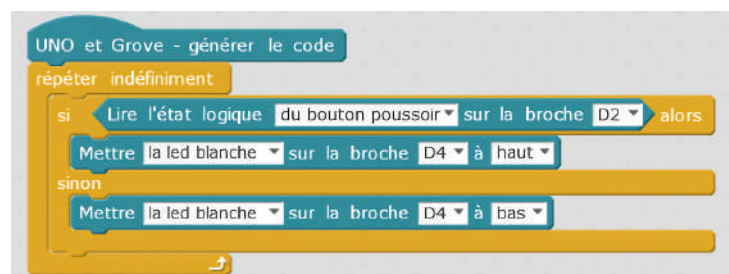


On accède alors à cet écran.



À gauche, on reconnaît le langage *Scratch/mBlock*, tandis qu'à droite, on visualise le même programme, exprimé en langage C (langage natif de la carte *Arduino*).

2. Remplacer l'instruction « drapeau vert » par son instruction équivalente pour ce mode autonome: l'instruction « Uno et Grove – générer le code ». Le programme devient alors :



3. À la place de l'ancienne commande « mettre à jour le microprogramme » (utile pour le mode connecté), on utilise la commande « téléverser dans *Arduino* » (en haut du programme écrit en langage C).

4. Cette opération (qui dure une dizaine de secondes) traduit le programme *mBlock* en langage C et l'importe dans la carte *Arduino*. Dès cet instant, la carte peut être déconnectée de l'ordinateur: elle est autonome! Attention: penser à mettre une alimentation (pile 9V par exemple), car la carte ne sera plus alimentée par le port USB de l'ordinateur.

5. Si l'on souhaite revenir au mode « connecté », il suffit de cliquer sur « retour », au-dessus du programme écrit en langage C.

Les élèves s'entraînent à réaliser ces différentes opérations, autant de fois que nécessaire pour que cela devienne un automatisme.

Recherche (par groupes) : réalisation de la maquette

La réalisation de la maquette globale peut se faire de 3 façons :

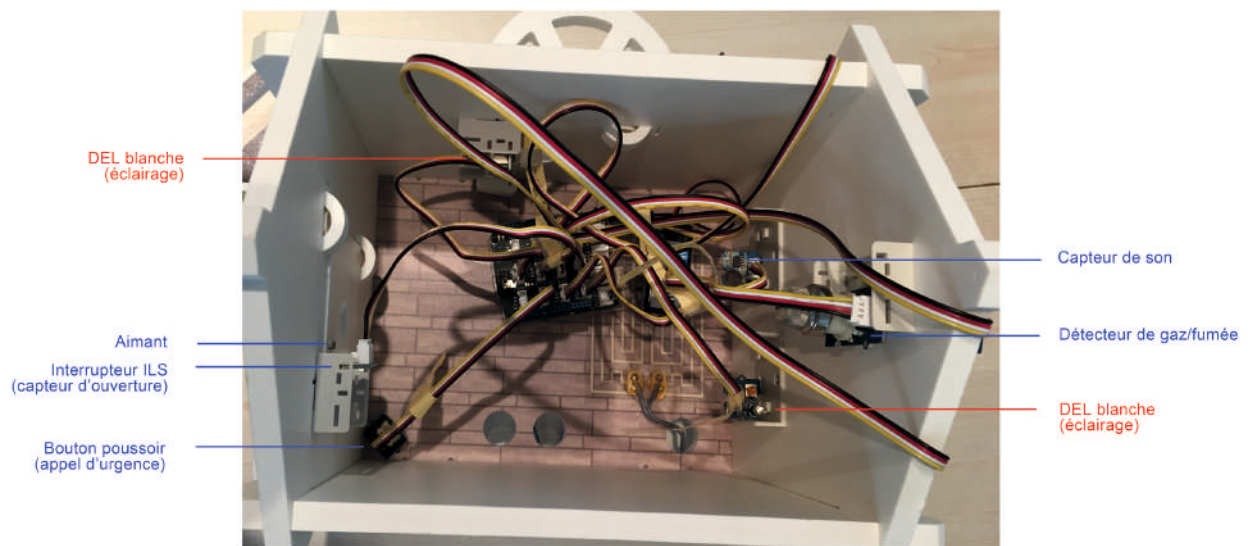
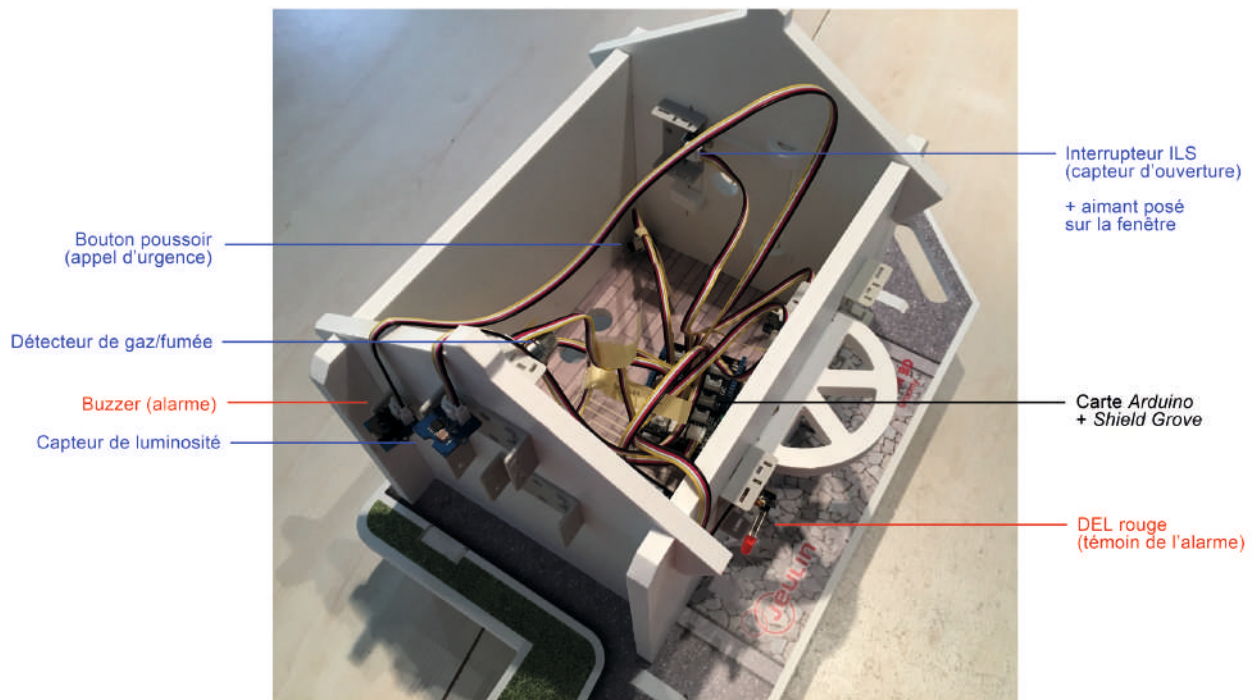
- Option 1 : chaque groupe réalise sa propre maquette. Les possibilités de chaque carte *Arduino* sont limitées à 4 entrées analogiques et 7 entrées/sorties digitales... ce qui limite d'autant les possibilités de branchement. Chaque maquette sera donc une version simplifiée de la maison domotique, car toutes les fonctions ne pourront pas être mises en œuvre.
- Option 2 : les groupes se rassemblent 2 par 2 pour réaliser une maquette commune. Chaque maquette peut donc être pilotée à l'aide de 2 cartes *Arduino* et posséder 2 fois plus de capteurs/ actionneurs que pour la solution précédente. Attention, chaque carte aura son programme propre!
- Option 3 : le professeur fabrique une grosse maquette, et la classe entière indique à quelques élèves comment faire les différents branchements. Cette option peut servir de restitution collective.

Notes pédagogiques

- L'option 2 a le mérite de faire collaborer les élèves à plus grande échelle et de leur faire produire une maquette plus complète/réaliste, et plus riche d'un point de vue fonctionnel. Cependant, il faudra s'assurer que tous les élèves sont bien actifs, ce qui est plus compliqué dans un grand groupe que dans un petit...
- Nous conseillons de réaliser tous les branchements et programmes en mode connecté avant l'intégration de l'ensemble dans la maquette, et de n'utiliser le mode autonome qu'en phase finale, quand tout fonctionne correctement.

Quelle que soit l'option choisie, il faut garder en tête la nécessité de prendre des photos des différentes étapes de réalisation de la maquette, voire, si possible, de courtes vidéos de cette maquette en fonctionnement.

Exemple de réalisation : la maquette ci-dessous n'utilise qu'une seule carte *Arduino* (option 1) et présente donc quelques limitations fonctionnelles. Néanmoins, elle est déjà suffisamment aboutie pour satisfaire les élèves.



Exemple de maquette fonctionnelle : la carte *Arduino* et son *Shield Grove* sont indiqués en noir ; les capteurs sont indiqués en bleu ; enfin, les actionneurs sont indiqués en rouge.

Bilan et conclusion

Le professeur organise une discussion collective permettant d'expliciter les notions acquises au cours de cette séance, notamment le fait que la carte *Arduino* peut être utilisée en mode « connecté » (branchée à l'ordinateur) ou en mode « autonome » (débranchée). Le mode autonome permet de fabriquer un véritable « système embarqué ».

Il explique ce qu'est un système embarqué : il s'agit d'un système électronique comprenant aussi bien les composants matériels que logiciels, destiné à exécuter une tâche précise. Ce système fonctionne en autonomie, ce qui lui impose certaines contraintes, comme par exemple une alimentation énergétique propre, une grande fiabilité matérielle et logicielle (car les systèmes embarqués sont supposés fonctionner sans intervention humaine), etc. Exemples de systèmes embarqués dans la vie quotidienne : système d'alarme, contrôleur de climatisation ou d'ascenseur, distributeur de billets automatique...

Les élèves notent ces conclusions dans leur cahier de projet, et n'oublient pas d'archiver les photos ou vidéos prises lors de la fabrication de la maquette, qui seront utilisées dans la dernière séance.

Prolongements possibles

Si la classe souhaite consacrer davantage de temps à ce projet, il est toujours possible d'améliorer/compléter une telle maquette. Voici quelques pistes :

- Permettre d'activer ou désactiver l'alarme en entrant un code (soit à l'aide du clavier, soit à l'aide d'une interface externe, comme un « makey makey » par exemple). Cela ne nécessite pas de capteur *Arduino* supplémentaire.
- Permettre d'activer/désactiver l'alarme à l'aide d'une carte (code couleur) : cela nécessite un capteur supplémentaire, compatible *Grove* (détection de couleur)
- Permettre la détection d'une chute, par exemple pour personnes âgées : cela nécessite un second capteur de vibration.
- Dessiner le schéma électrique de la maison : éclairage, interrupteurs, capteurs, actionneurs...
- Élargir le projet à la domotique en général, et pas uniquement à la question de la sécurité. Par exemple, on peut vouloir travailler sur la maîtrise de l'énergie (gestion de l'éclairage, du chauffage, des ouvertures...). Un tel projet nécessite d'autres capteurs/actionneurs. À l'aide d'un relai et d'une résistance chauffante, on peut piloter la température de la maison ; à l'aide de moteurs et de systèmes de transmission de mouvement, on peut piloter l'ouverture et la fermeture de volets ou fenêtres... Le projet peut rapidement devenir très ambitieux !



Séance 7 – Débat philo : les enjeux sociétaux de la domotique

Discipline dominante	Atelier philo
Résumé	Les élèves participent à un « atelier philo » portant sur les enjeux actuels de la domotique, et plus généralement des objets connectés. Facilitent-ils vraiment la vie des utilisateurs ? Permettent-ils de réduire l'isolement de chacun ? Présentent-ils des risques pour la sécurité ?
Notions (cf. scénario conceptuel, page 195)	Enjeux sociétaux : <ul style="list-style-type: none">• Les objets connectés peuvent être d'une grande aide au quotidien.• Les récentes révélations d'écoutes ou de piratages à très grande échelle mobilisent des citoyens inquiets de leur sécurité et du respect de leur vie privée.
Type d'activité	Séance débranchée
Matériel	Pour chaque binôme : <ul style="list-style-type: none">• Au choix, la Fiche 6, page 238 ou la Fiche 7, page 239

Préparation de l'atelier philosophique

Cette séance se déroule de façon très particulière : le premier tiers reprend une étude documentaire qui permet de conclure de façon générale le travail accompli jusqu'à présent sur la maison domotique. La seconde partie de la séance, quant à elle, propose d'aller plus loin dans la réflexion, sous la forme d'un « atelier philo ». Le professeur doit donc préparer sa séance et sa salle en conséquence. Idéalement, il est assisté du professeur documentaliste qui, en plus de guider les élèves dans leurs recherches documentaires, co-anime le débat. Sa position « extérieure » peut aider à l'expression de l'opinion de chacun et garantir le respect de cette opinion.

L'objectif d'une telle séance n'est en effet pas d'aboutir à une conclusion tranchée que chacun sera tenu d'adopter, mais de faire prendre conscience aux élèves de la complexité du dilemme auquel sont confrontées nos sociétés actuelles et de leur permettre de se forger une opinion argumentée.

Notes pédagogiques

- Différentes méthodes existent pour mener un atelier philosophique en classe. Celle qui est utilisée s'inspire de la méthode AGSAS-Lévine© qui est présentée dans l'ouvrage *L'enfant philosophe, avenir de l'humanité ?* de Jacques Lévine paru chez ESF Éditeur.
- Pour la préparation de cette séance, les deux professeurs auront sélectionné quelques ressources documentaires (notamment articles de presse) afin de nourrir la seconde étape du travail. Les sources ne manquent pas. En 2016, par exemple, une attaque informatique de très grande ampleur s'est appuyée sur des millions d'objets connectés (souvent domotiques) non sécurisés. Par ailleurs, de nombreux chercheurs ont montré qu'il était très facile d'accéder à des données confidentielles (par exemple, les images filmées par les webcams ou jouets connectés chez les particuliers), voire de piloter les appareils.

L'organisation de la salle doit refléter le caractère particulier de cette séance :

- Les chaises sont placées en cercle de façon à aménager un espace de parole (les tables sont repoussées à l'extérieur du cercle, contre les murs de la salle).
- Les élèves prennent place sur ces chaises, dans le cercle.
- Les professeurs se placent en dehors du cercle : ils n'interviennent pas au cours de l'atelier.

Situation déclenchante

Le professeur distribue aléatoirement une Fiche 6 ou une Fiche 7 aux différents binômes. Il leur demande d'identifier les objets pris en photo, et de deviner leur fonction.

Notes scientifiques

Voici les spécificités des robots choisis pour illustrer la Fiche 6 et la Fiche 7 :

- Les bras mécaniques disposent de capteurs pour vérifier la justesse de leur geste et leurs niveaux de consommables.
- Baxter est doté d'une reconnaissance de formes pour savoir quels objets récupérer sur un tapis roulant.
- BigDog adapte sa démarche au terrain pour continuer d'avancer malgré les obstacles.
- En groupes, les Eporo imitent les bancs de poissons pour rouler de concert, sans embouteillage ni accident.
- Les robots aident les scientifiques à explorer les mécanismes du déplacement : le Harvard Ambulatory MicroRobot pour la marche à plusieurs pattes (existe en version mille-pattes), le Honda P2 pour la marche bipède, Robobee pour le vol, le poisson G9 pour la nage...
- Han explore la reconnaissance et la reproduction des émotions par les mouvements subtils du visage.
- Roomba est un aspirateur qui visite de lui-même la pièce et repart se recharger quand ses batteries s'épuisent
- Thymio, petit robot éducatif, utilisé pour le projet « robotique », page 300.
- *Sojourner* fait partie d'une longue série de robots explorateurs du système solaire (le premier fut *Lunokhod 1*, envoyé sur la Lune en 1970).

Rapidement, les élèves comprennent qu'il s'agit exclusivement de robots. L'enseignant demande de justifier cette affirmation, ce qui n'est pas aisé.

La définition d'un robot est : *une machine capable d'interagir avec son environnement ; cela suppose qu'elle dispose donc de capteurs pour détecter son environnement, d'actionneurs pour agir sur lui, et d'un ordinateur pour savoir comment agir.*

En classe entière, les élèves regardent les photographies à la recherche de ces trois ingrédients. Les actionneurs sont faciles à repérer : roues, engrenages, ailes, nageoires, pattes... Les programmes sont évidemment invisibles. Les capteurs, quant à eux, sont difficiles à repérer et encore moins faciles à identifier : on espère que les yeux de Han, d'Eporo ou G9 sont bien des capteurs, mais rien n'est moins sûr ; quelques capteurs de Thymio se voient bien (les 5 petites fenêtres noires sur sa face avant) mais ne sont pas identifiables en tant que tels, etc.

L'enseignant ne laisse pas cet exercice s'éterniser, et il propose rapidement de comparer ces robots, car ce sont bel et bien des robots, à leur maison domotique. *Une maison domotique est-elle un robot ?* Les élèves réalisent vite qu'ils ont inclus dans leur maquette des capteurs (lumière, son, gaz, etc.), des

actionneurs (alarme, DEL, etc.), et des programmes, parfois embarqués et autonomes. Oui, disent-ils, d'après cette définition, la maison domotique est un robot.

Le professeur peut alors affiner cette conclusion : *un robot est souvent un objet de taille modérée et qui comporte un « corps » bien déterminé. Dans le cas de la maison domotique, le corps du robot n'est pas d'une seule pièce : les scientifiques parlent plutôt de « comportement robotique ». C'est analogue à un robot, sans en être un véritablement.*

Rappel des règles de l'« atelier philo »

Que les élèves aient déjà expérimenté ce type de travail ou non, le professeur explique maintenant (ou fait rappeler) aux élèves le déroulement et les règles inhérentes à l'« atelier philo » qui vient :

- Le professeur énonce le thème de l'atelier et chacun réfléchit pendant une minute à l'avis qu'il a sur cette question. Puis, les élèves disposent de 10 minutes pour s'exprimer sur le sujet, temps au cours duquel le professeur ne prend pas la parole. Un dispositif d'enregistrement est placé au centre du cercle car l'« atelier philo » sera ensuite retranscrit (sans indiquer les prénoms des personnes qui s'expriment) et distribué à chacun.
- Les élèves sont libres de dire ce qui leur passe par la tête, tout en essayant de prendre de la hauteur par rapport au sujet. Il n'y a pas de bonne ou de mauvaise réponse. L'« atelier philo » est l'occasion de penser par soi-même, d'observer le cheminement individuel et collectif de la pensée.
- On ne peut s'exprimer que lorsqu'on a le « bâton de parole ». Celui-ci passe de main en main et il n'est pas obligatoire de dire quelque chose quand il arrive jusqu'à soi. Si l'élève n'a pas envie de parler, il transmet le bâton de parole à son voisin.
- Une des règles fondamentales est le respect de la parole d'autrui. On a le droit de ne pas être d'accord avec quelqu'un, mais on n'a pas le droit de se moquer ou de juger ce qui a été formulé, ni de parler avec vulgarité. On doit écouter ce que disent les autres.

Une fois ces règles énoncées, le professeur peut demander si certains élèves ne se sentent pas capables de les respecter, auquel cas ils sont priés de sortir de l'espace de parole. Ils ne seront alors pas autorisés à intervenir pendant l'atelier.

Déroulement de l'atelier

Le professeur annonce le thème de l'atelier : *voudriez-vous habiter une maison domotique, et pourquoi ?* Pendant que les élèves réfléchissent quelques instants, le dispositif d'enregistrement audio est déclenché. Le professeur demande qui veut commencer et donne le « bâton de parole » à l'élève qui souhaite s'exprimer en premier, puis sort de l'espace de parole. Il peut aussi, s'il le souhaite, donner ce bâton de parole à un élève au hasard.

Le bâton passe de main en main. Chacun parle à son tour s'il le désire jusqu'à ce que le temps imparti soit écoulé.

À la fin des 10 minutes, l'enseignant demande si les élèves qui ne se sont pas exprimés pendant l'atelier souhaitent le faire, puis si certains veulent donner leur ressenti par rapport au déroulement de l'atelier : qualité de l'écoute, intérêt de ce qui a été formulé, etc.

Conclusion : examen des arguments du débat public sur les objets connectés

Après le déroulement de l'atelier proprement dit, les professeurs reviennent dans le cercle et proposent aux élèves d'écouter l'enregistrement.

Les professeurs proposent aux élèves de lire quelques documents choisis à l'avance (coupures de presse, voir la note pédagogique en début de séance...).

Voici un recueil des arguments les plus fréquemment rencontrés dans le débat public sur la domotique (éviter de trop élargir le débat même si, on s'en rendra compte à la fin, ce qui sera dit vaudra sans doute pour l'informatique embarquée en général).

- Avantages :

- Facilite la vie des particuliers : robots aspirateurs, grille-pain intelligent, chauffage de la salle de bains qui se met en route quelques minutes avant le réveil, etc.

- Télésurveillance : on peut surveiller sa maison à distance, ou faire qu'elle se surveille toute seule.

- Sécurité : les personnes âgées, souvent seules, peuvent plus facilement appeler de l'aide en cas de besoin.

- Accessibilité : les personnes handicapées peuvent trouver de l'aide dans certaines tâches du quotidien (ouvrir les volets...)

- Inconvénients :

- Coût : une maison domotique est plus chère qu'une maison classique.

- Résilience : que devient-on en cas de coupure électrique ou en cas de panne d'un élément du système ?

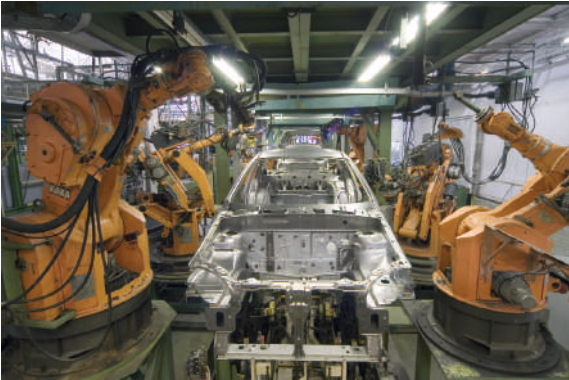
- Isolement : avant, on sympathisait avec ses voisins, qui pouvait nous rendre les mêmes services (allumer le chauffage le jour de notre retour de vacances, nous aider en cas d'accident...); l'autonomie totale ne risque-t-elle pas d'isoler les personnes les unes des autres ?

- Confidentialité : que deviennent toutes ces données ? Les systèmes embarqués sont souvent connectés à Internet : cela permet de suivre à distance ce qui s'y passe, mais cela pose la question de la confidentialité. Qui a envie qu'une personne externe accède aux images filmées par sa webcam ? à ses heures d'entrée et de sortie ? à l'identité de ses interlocuteurs ? au contenu de son réfrigérateur ?

- Sécurité : les appareils domotiques sont parmi les moins sécurisés, et sont à l'origine de nombreux actes de piratage de grande ampleur.

FICHE 6

Quel est le point commun entre ces objets? (1/2)



Bras mécaniques



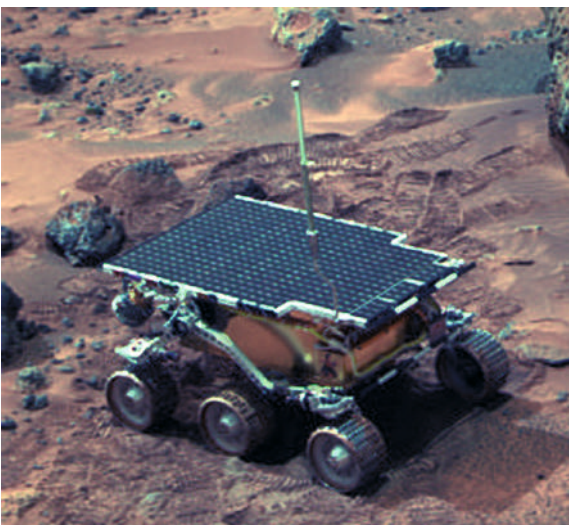
Roomba©



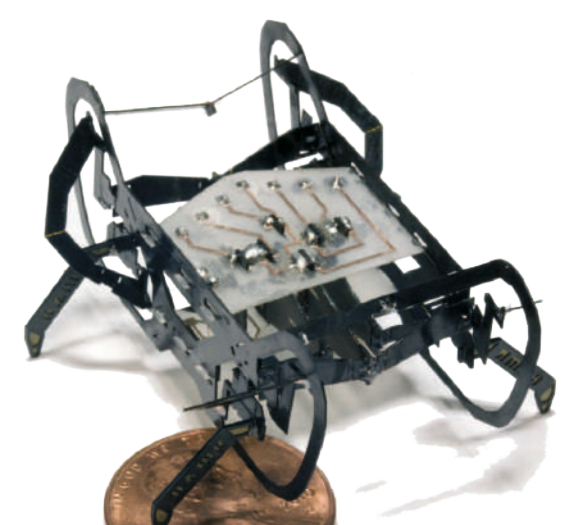
Han



Eporo©

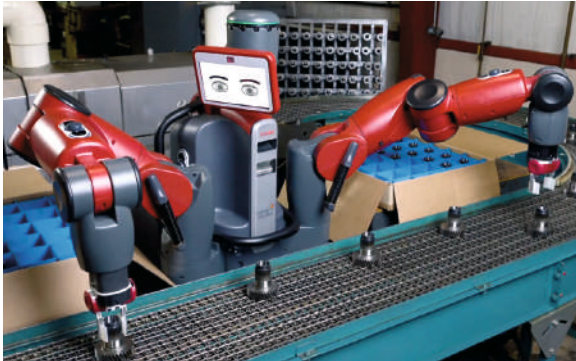


Sojourner©



HAMR (© Harvard University)

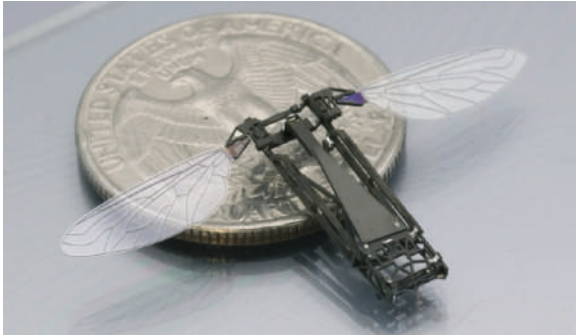
FICHE 7
Quel est le point commun entre ces objets? (2/2)



Baxter©



Poisson G9



Robobee©



Thymio



BigDog©



Honda-P2©



Séance 8 – Restitution du projet

Discipline dominante	Atelier philo
Résumé	Les élèves réalisent un diaporama (ou une vidéo de type making of) destiné à présenter leur projet (aux autres professeurs, aux autres classes, aux familles...).
Type d'activité	TICE
Matériel	Pour chaque groupe : <ul style="list-style-type: none">• Un ordinateur équipé d'un logiciel de diaporama (<i>OpenOffice Presentation, MS Powerpoint...</i>)• Un appareil photo numérique• L'ensemble des documents rassemblés tout au long du projet (photos, notes, vidéos...) Pour la classe : <ul style="list-style-type: none">• Un vidéoprojecteur

Note pédagogique

La plupart des élèves tireront bénéfice à présenter leur projet à un tiers: ce sera l'occasion pour eux de revenir sur l'objectif initial, les étapes traversées, les solutions trouvées... Un tel travail offre une belle occasion de structurer les connaissances acquises précédemment.

Réalisation d'un diaporama ou d'un making of (par groupes)

Le professeur explique qu'une bonne façon de structurer et de stabiliser ses connaissances consiste à expliquer ce qu'on a fait à une tierce personne.

Il propose donc à chaque groupe de construire un diaporama ou une vidéo qui présente l'ensemble du projet:

- Objectif initial
- Étapes de réalisation: cahier des charges, premiers essais de branchement et de programmation, mise en œuvre de fonctions plus complexes, construction de la maquette complète
- Les connaissances et compétences acquises au cours du projet: au sujet de la robotique, de la programmation, de la logique, des enjeux sociétaux ou, plus généralement, des compétences transversales (autonomie, collaboration, expression...)

Restitution finale

Nous pensons qu'il est peu intéressant que les groupes restituent leur travail auprès de leurs pairs, étant donné que tout le monde, dans la classe, sait ce qui a été fait (le projet a été ponctué de nombreuses mises en commun). Une telle restitution interne peut toutefois se faire en demi-classe afin de discuter de la forme de cette restitution (qualité du diaporama, qualité de l'expression orale) et de permettre aux groupes de s'améliorer avant une restitution externe (voir ci-après).

Il est bien plus intéressant de restituer ce travail auprès de personnes qui n'ont pas suivi le projet ! Les possibilités sont nombreuses : autres professeurs (en particulier, le professeur de mathématiques, lui aussi concerné par les apprentissages en informatique, ou le professeur principal), autres classes, parents (à l'occasion d'une journée portes-ouvertes)...

Cette ressource est issue du projet thématique **1,2,3... CODEZ !**, paru aux Éditions Le Pommier.

Claire Calmet, Mathieu Hirtzig et David Wilgenbus

1,2,3... CODEZ !

Enseigner l'informatique à l'école et au collège
(cycles 1, 2 et 3)

FONDATION
La main à la pâte
POUR L'ÉDUCATION À LA SCIENCE

Qu'il s'agisse de préparer les enfants aux métiers de demain, de les aider à comprendre le monde numérique dans lequel ils vivent – afin qu'ils soient en mesure d'agir sur lui et non de le subir –, de les sensibiliser aux enjeux de citoyenneté, ou encore de favoriser la coopération ou développer leur créativité... l'informatique doit être enseignée à tous, dès le plus jeune âge.

Le projet « 1, 2, 3... codez ! » développé par la Fondation *La main à la pâte*, Inria et France 101 vise à initier les élèves et leurs enseignants à la science informatique, de la maternelle à la classe de 6^e. Il propose à la fois des activités branchées (nécessitant un ordinateur, une tablette ou un robot) permettant d'introduire les bases de la programmation et des activités débranchées (informatique sans ordinateur) permettant d'aborder des concepts de base de la science informatique (algorithme, langage, information...).

Un outil clés en main
Ce guide pédagogique comporte :

- 3 progressions pour la classe (cycles 1, 2 et 3)
 - Une approche pluridisciplinaire associant démarche d'investigation et pédagogie de projet ;
 - Des séances clés en main, testées en classe, organisées en séquences thématiques et scénarisées pour chaque cycle ;
 - Des fiches documentaires à photocopier ;
- Des éclairages pédagogiques et scientifiques pour guider l'enseignant dans la mise en œuvre du projet.

Les auteurs
Claire Calmet est formatrice et responsable des liens avec le monde de l'entreprise et de la recherche à la Fondation *La main à la pâte*.
Mathieu Hirtzig est webmestre et médiateur scientifique à la Fondation *La main à la pâte*.
David Wilgenbus est formateur et responsable de la production de ressources à la fondation *La main à la pâte*. Il coordonne le projet « 1, 2, 3... codez ! ».

FONDATION
La main à la pâte

Lancée en 1996 par Georges Charpak, prix Nobel de physique, avec le soutien de l'Académie des sciences et du ministère de l'Éducation nationale, *La main à la pâte* vise à promouvoir à l'école primaire un enseignement de science et de technologie de qualité : <http://www.fondation-lamap.org>

Avec le soutien de :

Illustration : Catherine Zimmmermann

Éditions
Le Pommier

74651106
21 €
Diffusion Bélin

Retrouvez l'intégralité de ce projet sur : <https://www.fondation-lamap.org/projets-thematiques>.

Fondation *La main à la pâte*

43 rue de Rennes
75006 Paris
01 85 08 71 79
contact@fondation-lamap.org

Site : www.fondation-lamap.org

FONDATION
La main à la pâte
POUR L'ÉDUCATION À LA SCIENCE