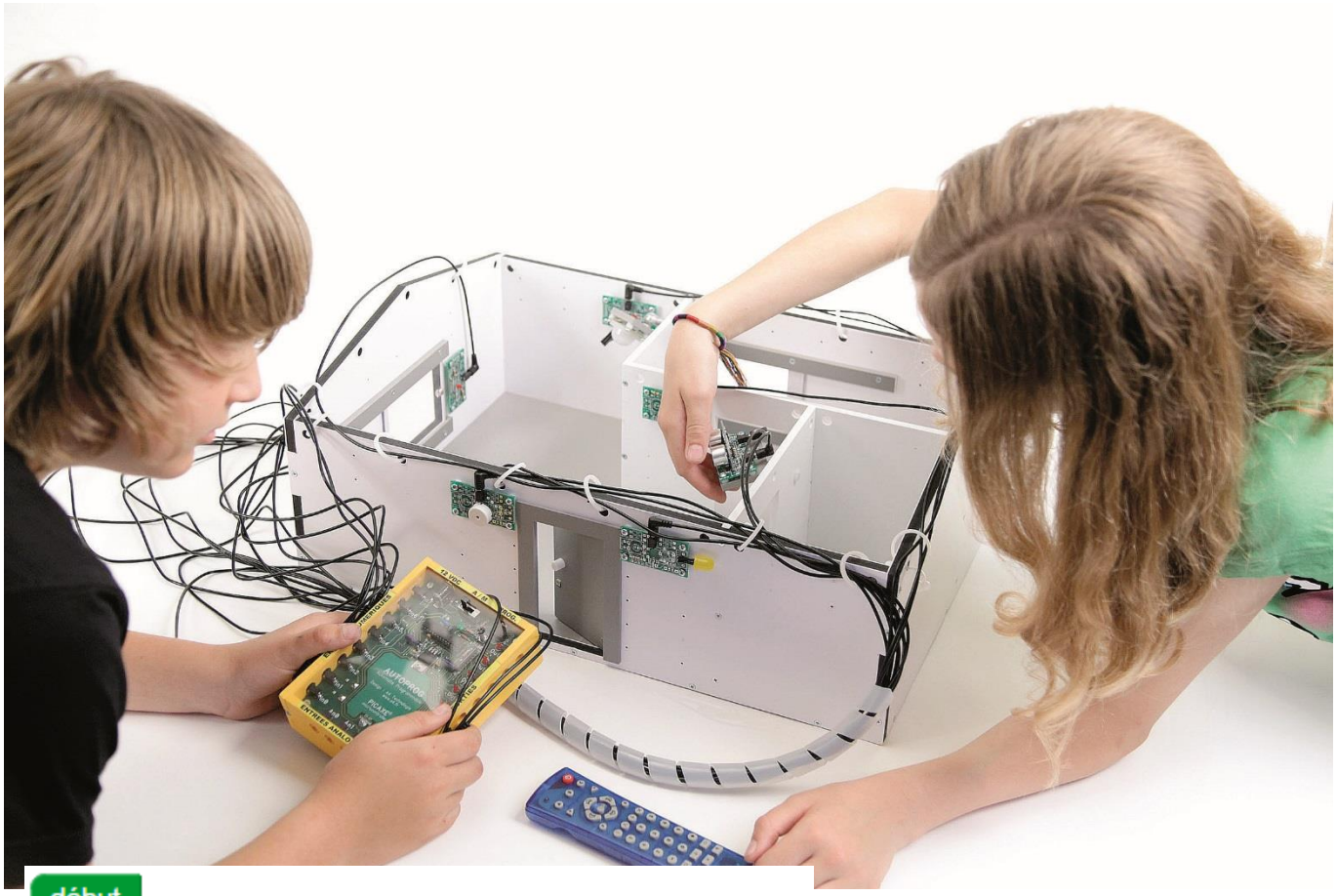
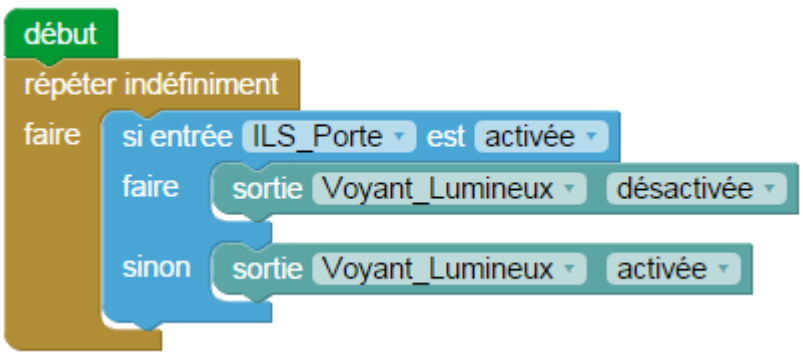
AutoAlarme

Maquette programmable
avec PICAXE Editor / Blockly pour PICAXE





TECHNOLOGIE

Ressources disponibles pour le projet AutoAlarme

Autour du projet AutoAlarme, nous vous proposons un ensemble de **ressources téléchargeables gratuitement sur le wiki.**

AutoAlarme

- Fichiers **3D** (SolidWorks, Edrawings et Parasolid) de la maquette et de ses options.
- Dossier **technique** AutoLumi pour la mise en œuvre de la maquette ;
- Une notice d'utilisation de l'**option Bluetooth** ;

Logiciels Picaxe Editor 6 / Blockly et App Inventor

- Procédure d'installation du driver pour le câble de programmation.
- Manuel d'utilisation Picaxe Editor 6.
- Notice d'utilisation App Inventor 2.

Activités / Programmation

- Fichiers modèles et fichiers de correction des programmes pour Picaxe EDITOR 6 (organigrammes et blocs) et AppInventor.

NOTE : Certains fichiers sont donnés sous forme de fichier.zip.

Logiciels, programmes, manuels utilisateurs téléchargeables gratuitement sur www.a4.fr

SOMMAIRE

Introduction

AutoAlarme

La maquette système d'alarme (BE-AALAR) est une reproduction homothétique d'un système d'alarme automatisé réel : plusieurs étages, capteurs fin de course, contrepoids, moteurs, etc. Programmable et pilotée par les systèmes AutoProgX2 ou AutoProgUno, elle permet une activité de programmation complète par rapport aux attendus de fin de cycle collège : l'algorithmique en maths, l'étude de scénarios, la programmation et la mise en œuvre en Technologie.

Vous trouverez dans ce document tout le nécessaire pour démarrer des activités de programmation autour du système d'alarme :

- La mise en œuvre de la maquette : câblage et configuration des modules.
- Différents scénarios de programmation, du plus simple au plus complexe, avec des exemples de programmes tout faits en langage par blocs.
- Des exercices complémentaires pour les différents modules en option : télécommande infrarouge et module Bluetooth.

Les environnements de programmation graphique

Tous les programmes correspondant aux activités menées autour de la maquette AutoAlarme ont été réalisés sous **PICAXE Editor 6**. En effet, ce logiciel de programmation graphique présente plusieurs **avantages** :

- Gratuit
- Blocs et organigrammes (proche algorigrammes).
- Personnalisation des noms des entrées/sorties.
- Personnalisation du jeu d'instructions.
- Mode de simulation visuelle à l'écran pour mettre au point et débugger les programmes.

Vous pouvez aussi utiliser **Blockly for Picaxe** : environnement de programmation par blocs simplifié (nombre de menus limité et personnalisation des entrées/sorties non disponibles).

Pour les activités menées avec un smartphone ou une tablette, les programmes et applications ont été réalisés sous **App Inventor 2**.
Il s'agit d'un environnement de développement pour concevoir des applications pour smartphone ou tablette Android.
Il a été développé par le MIT pour l'éducation. Il est gratuit et fonctionne via internet avec Blockly.

Le dossier

Ce document propose un parcours progressif pour découvrir et se perfectionner avec la programmation en se basant sur une série d'exemples ludiques autour de la maquette AutoAlarme grâce à ses capteurs et actionneurs.
Il est organisé en fonction des niveaux de programmation.

Niveau 1 :
Découverte progressive du jeu d'instructions et des fonctionnalités de base de la maquette et maîtrise des principes fondamentaux pour concevoir un programme : séquences, boucles, structures conditionnelles (test) et variables.

Niveau 2 :
Approfondissement des principes de programmation abordés dans le niveau 1 en concevant des programmes plus élaborés qui répondent à des cas concrets d'utilisation de la maquette (version de base).

Niveau 3 :
Exemples d'utilisation des différentes options proposées : télécommande infrarouge et module Bluetooth.

Les fiches exercices

Pour chaque niveau de programmation, nous vous proposons des fiches exercices avec :
- un objectif : ce que doit faire le programme ;
- un fichier modèle : un programme vide avec un jeu d'instructions limité (suffisant pour réaliser l'exercice) ;
- un fichier de correction qui propose un exemple de programme réalisé sous Picaxe Editor 6 en blocs (extension .xml) et en organigrammes pour le niveau 1 uniquement (extension .plf).

Intérêt du fichier modèle :
- il évite aux utilisateurs de se perdre dans une multitude d'instructions ;
- il limite les propositions possibles ;
- il facilite la correction et l'analyse des erreurs.

Deux approches :
- Avec les exemples de programmes, les utilisateurs découvrent les principes de la programmation graphique en organigrammes ou en blocs : chargement d'un programme, modification d'un programme et vérification sur le matériel (ex : modification des temps d'attente, etc.).
- Les utilisateurs conçoivent eux-mêmes le programme pour atteindre l'objectif proposé, en organigrammes ou en blocs (à partir du fichier modèle). Ils peuvent ensuite le comparer au fichier de correction.

Principe de nommage des fichiers :

- **AL** : pour AutoAlarme
- **N** : niveau de programmation 1-2-3
- **A-B-C** : jeu d'instructions du plus simple au plus avancé

Exemple : AL_N3_B2.xml
Correspond au niveau 3 avec le jeu d'instructions B, adapté aux objectifs « avancés » de ce niveau.

Prérequis

Pour la version de base :

- Installer le logiciel **Picaxe Editor 6** ou **Blockly for Picaxe :** http://www.picaxe.com/Software
- **Maquette** AutoAlarme (Réf. BE-AALAR).
- **Câble de programmation** Picaxe USB (Réf : CABLE-USBPICAXE).
- **Interface programmable** AutoProgX1 ou X2 (Réf. K-APV2).
- 11 **cordons de liaison** jack compatibles AutoProg pour établir les liaisons entre l'interface programmable et la maquette.

Pour l'option Bluetooth :

- **Tablette ou smartphone** Android 5 ou + équipés de Bluetooth V3.
- Connexion internet pour accéder à **App Inventor :** http://ai2.appinventor.mit.edu/
- Compte Gmail requis.

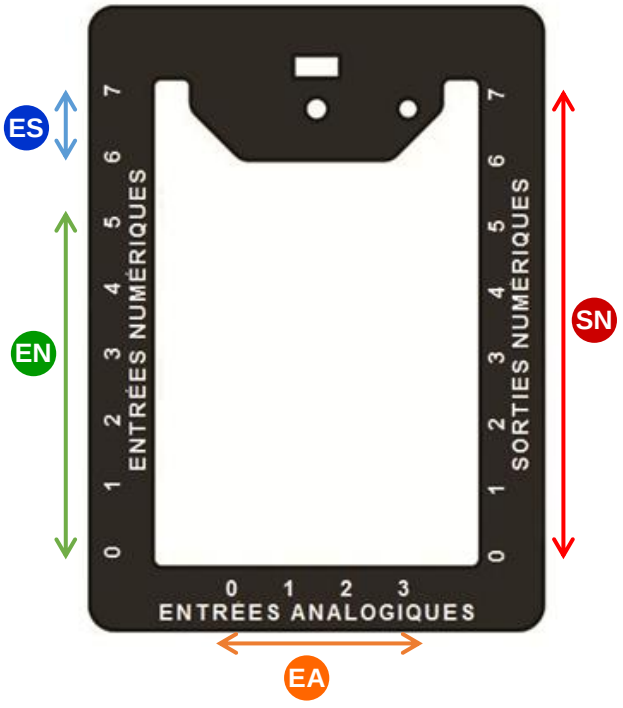
Caractéristiques techniques

Le guide de montage ainsi que les caractéristiques techniques des composants sont détaillés dans le dossier technique disponible sur le wiki.

Mise en œuvre d'AutoAlarme

Tableau d'affectation des entrées et sorties

	Module de communication pour entrées / sorties numériques	Broche Blockly	Etiquette Blockly
ES			
7	Communication Bluetooth envoi de données	C.7	BLTH_TX
6	Communication Bluetooth réception de données	C.6	BLTH_RX
EN	**Modules capteurs pour entrées numériques**		
5	Capteur à ultrasons	C.5	Capteur_Ultrason
4	Récepteur infrarouge	C.4	Recepteur_IR
3	Capteur PIR	C.3	Detection_PIR
2	Capteur magnétique de la porte d'entrée	C.2	ILS_Porte
1	Capteur magnétique de la fenêtre du salon	C.1	ILS_Salon
0	Capteur magnétique de la fenêtre de la cuisine	C.0	ILS_Cuisine
EA	**Modules capteurs pour entrées analogiques**		
3	*(libre)*	*A.3*	
2	*(libre)*	A.2	
1	*(libre)*	A.1	
0	*(libre)*	A.0	
SN	**Modules actionneurs sorties numériques**		
7	*(libre)*	B.7	
6	*(libre)*	B.6	
5	*(libre)*	B.5	
4	*(libre)*	B.4	
3	*(libre)*	B.3	
2	Emetteur infrarouge	B.2	Emetteur_IR
1	Borne reliée au buzzer	B.1	Buzzer
0	Module signal LED jaune	B.0	Voyant_Lumineux



Programmation version de base niveau 1

Objectifs :

- Découvrir et maîtriser le matériel avec des exemples très simples pour débuter en programmation.
- Appréhender les différentes fonctionnalités du matériel.

Ce niveau permet de découvrir toutes les fonctionnalités de base d'AutoAlarme, en apprenant les structures de base de la programmation. Et en particulier celles demandées dans les nouveaux programmes : séquences, boucles, structures conditionnelles et enfin les variables.

Nous vous conseillons pour chaque exercice d'essayer d'écrire le programme vous-même, en partant du modèle de base (fourni avec les exercices), avant de regarder la correction et l'explication de chaque programme.
Par exemple, pour le programme « AL_N1_A1.xml », charger le programme modèle « AL_Base.xml ».

Dans chaque programme modèle du niveau 1, vous trouverez la liste de blocs nécessaires à la réalisation des exercices des sous niveaux A, B, C et D.
Au fur et à mesure de l'avancement dans les sous niveaux, la liste de blocs s'agrandit jusqu'à retrouver tous les blocs nécessaires pour piloter complétement la maquette.

Nom du fichier	**Description**	**Objectif**
Niveau 1 A **Fichier modèle :** AL_N1_A.xml		
AL_N1_A1	allumer le voyant lumineux pendant 3 secondes puis l'éteindre.	Fonctionnalité matérielle abordée : -Allumage/extinction du voyant lumineux. Notions de programmation abordées : -séquence d'instructions -temps d'attente -boucle infinie
AL_N1_A2	allumer le voyant lumineux pendant 3 secondes puis l'éteindre, recommencer.	
AL_N1_A3	faire clignoter le voyant lumineux avec une période de 6 secondes indéfiniment.	
Niveau 1 B **Fichier modèle :** AL_N1_B.xml		
AL_N1_B1	Allumer un voyant lumineux lorsque la porte de devant est ouverte.	Fonctionnalité matérielle abordé : -Gestion des capteurs oui ou non. -Utilisation d'un buzzer. Notions de programmation abordées : -boucle qui dépend d'une entrée.
AL_N1_B2	Activer et désactiver un Buzzer toutes les 3 secondes pendant 1 seconde.	
AL_N1_B3	Créer une alarme qui sonne et allume le voyant lumineux toutes les 0.5 secondes.	
AL_N1_B4	Reprendre l'exercice précédent et rajouter les capteurs situés sur les fenêtres.	
AL_N1_B5	Reprendre l'exercice précédent et rajouter les capteurs situés sur les fenêtres.	
Niveau 1 C **Fichier modèle :** AL_N1_C.xml		
AL_N1_C1	Allumer le voyant lumineux lorsqu'il y a un obstacle entre le récepteur et l'émetteur infrarouge.	Fonctionnalité matérielle abordé : -Gestion de différents capteurs. Notions de programmation abordées : -Le test d'une entrée (si/sinon). -Lire une valeur.
AL_N1_C2	Récupérer la valeur de distance envoyée par un télémètre à ultrasons.	
AL_N1_C3	Allumer le voyant lumineux lorsqu'il y a un obstacle trop proche du télémètre à ultrasons.	
AL_N1_C4	Allumer le voyant lumineux lorsqu'il y a une détection sur le capteur PIR.	

Niveau 1 - A

Exercice niveau 1 - A.1 : Activer / désactiver un témoin lumineux

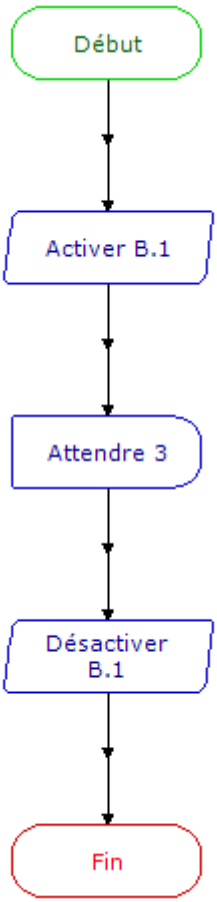
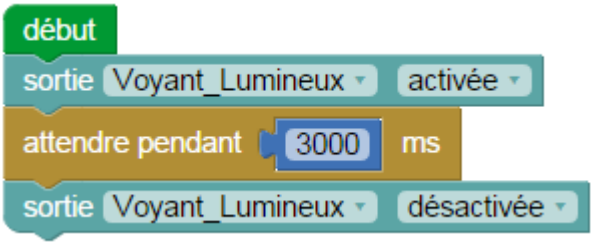
Fichier modèle : AL_BASE.xml

Objectif : allumer le voyant lumineux pendant 3 secondes puis l'éteindre.

Notions abordées : séquence d'instructions, activation / désactivation d'une sortie, temps d'attente.

Instructions utilisées :

Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_A1_Organigramme.plf	Fichier Blockly : AL_N1_A1.xml

Remarque : avec le langage de programmation par blocs la dernière instruction exécutée marque la fin du programme.

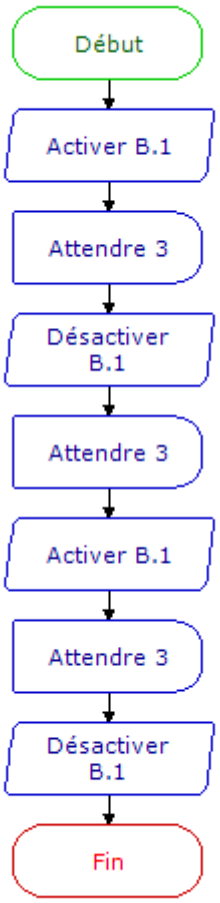
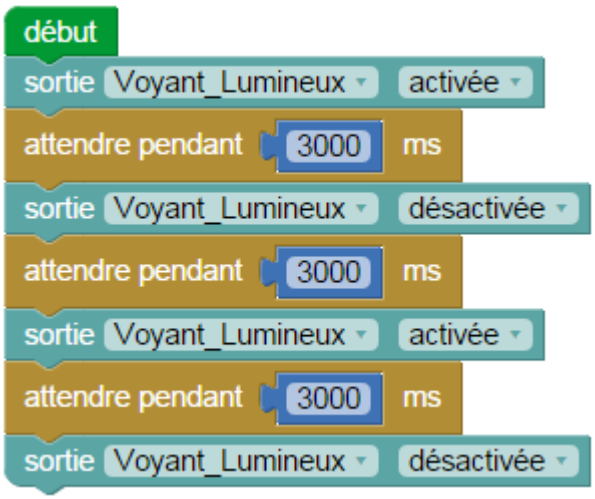
Exercice niveau 1 - A.2: Répéter une action deux fois

Objectif : allumer le voyant lumineux pendant 3 secondes puis l'éteindre, recommencer.

Notions abordées : séquence d'instructions, activation / désactivation d'une sortie, temps d'attente.

Instructions utilisées :

Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_A2_Organigramme.plf	Fichier Blockly : AL_N1_A2.xml

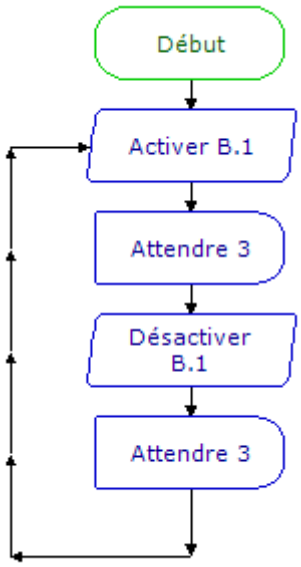
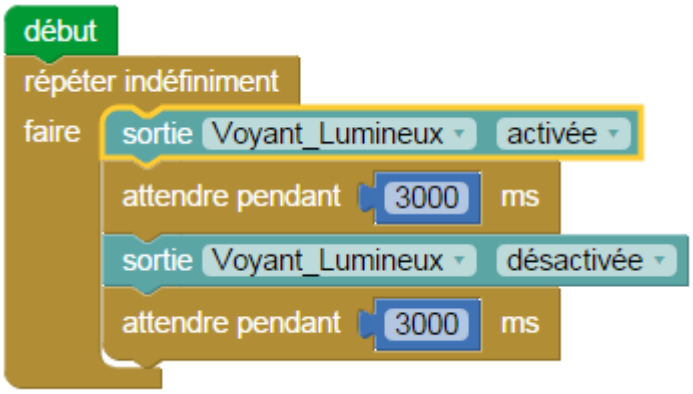
Exercice niveau 1 - A.3 : Répéter une séquence indéfiniment

Objectif : faire clignoter le voyant lumineux avec une période de 6 secondes indéfiniment.

Notion abordée : la boucle infinie.

Instructions utilisées :

Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_A3_Organigramme.plf	Fichier Blockly : AL_N1_A3.xml

Remarque : le programme ne peut s'arrêter lorsqu'il est dans une boucle infinie. Le seul moyen de sortir de la boucle est de faire un Reset ou d'éteindre et rallumer le boîtier AutoProg.

Niveau 1 - B

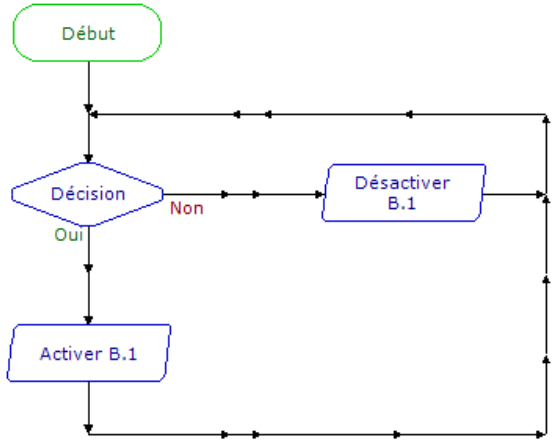
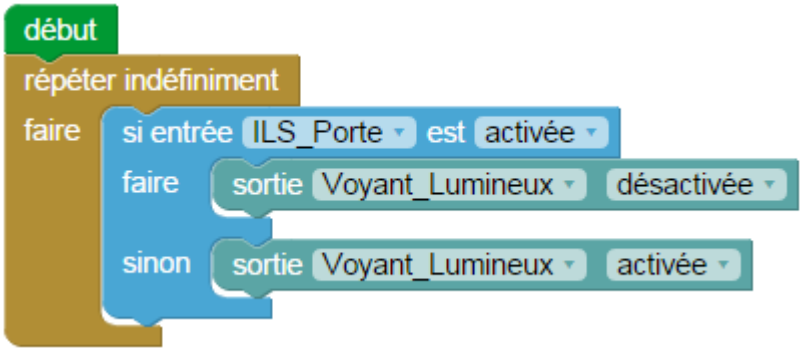
Exercice niveau 1 - B.1 : Utiliser un capteur magnétique

Objectif : Allumer un voyant lumineux lorsque la porte de devant est ouverte

Notion abordée : utilisation d'un capteur magnétique

Instructions utilisées :

Correction :

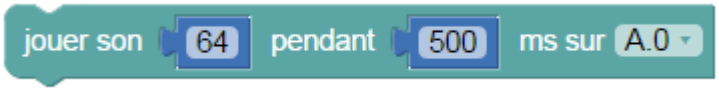
Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_B1_Organigramme.plf	Fichier Blockly : AL_N1_B1.xml

Exercice niveau 1 - B.2 : Utiliser un buzzer

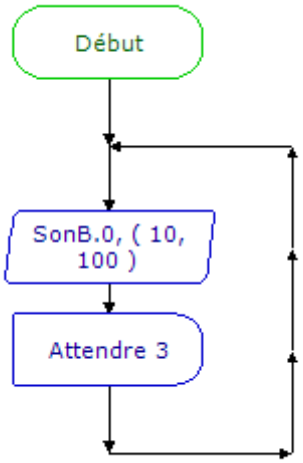
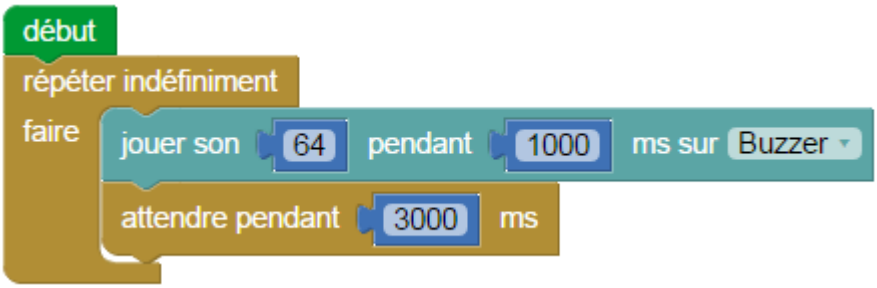
Objectif : Activer et désactiver un Buzzer toutes les 3 secondes pendant 1 seconde

Notion abordée : utilisation du buzzer

Instructions utilisées :



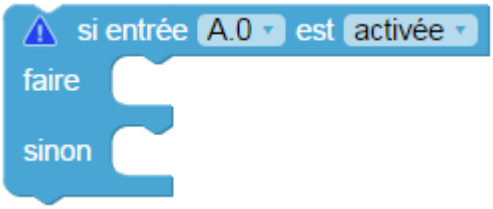
Correction :

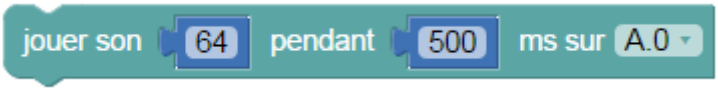
Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_B2_Organigramme.plf	Fichier Blockly : AL_N1_B2.xml

Exercice niveau 1 - B.3 : Première alarme

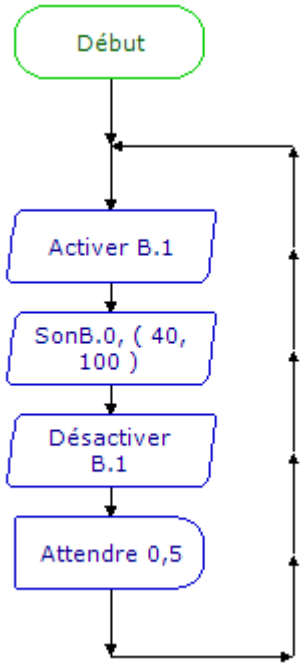
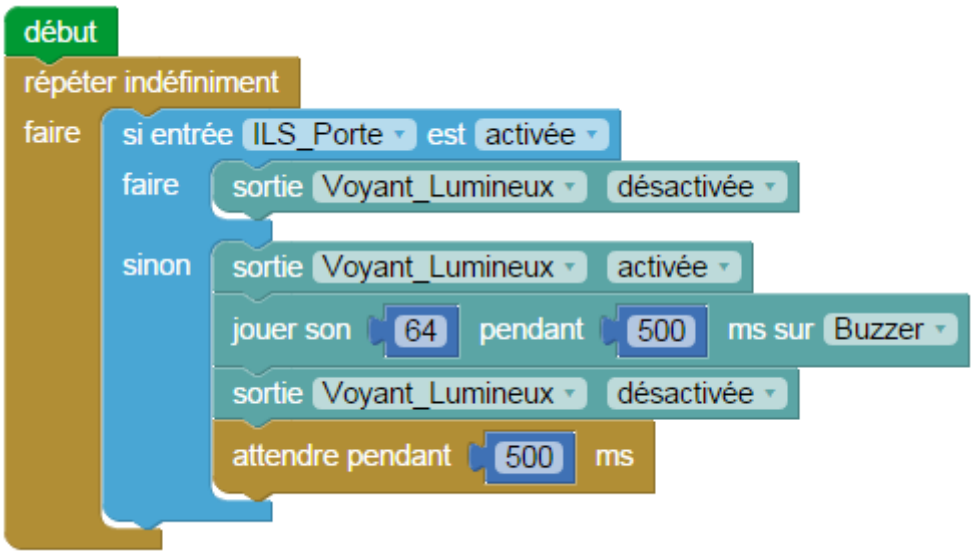
Objectif : Créer une alarme qui sonne et allume le voyant lumineux toutes les 0,5 secondes

Instructions utilisées :





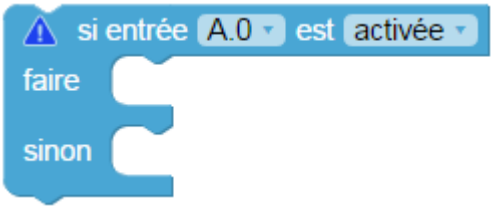
Correction :

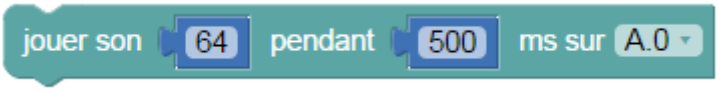
Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_B3_Organigramme.plf	Fichier Blockly : AL_N1_B3.xml

Exercice niveau 1 - B.4 : Activer l'alarme depuis plusieurs endroits

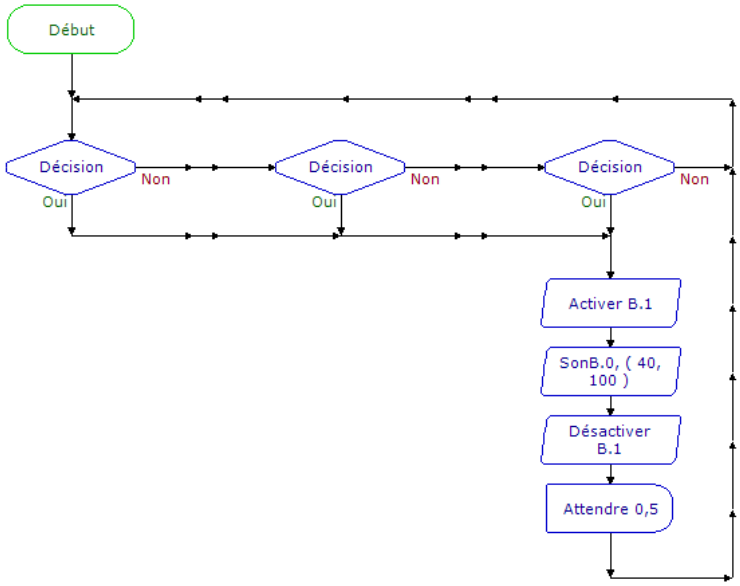
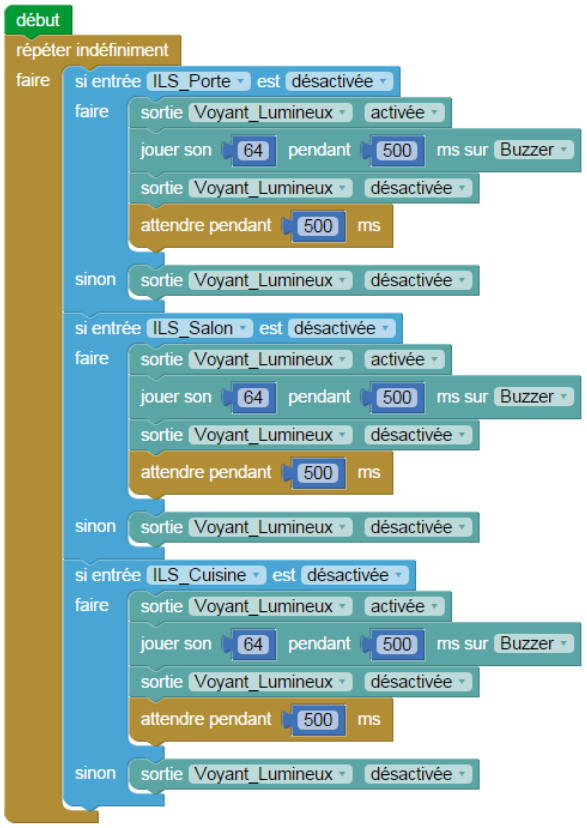
Objectif : Reprendre l'exercice précédent et rajouter les capteurs situés sur les fenêtres

Instructions utilisées :





Correction :

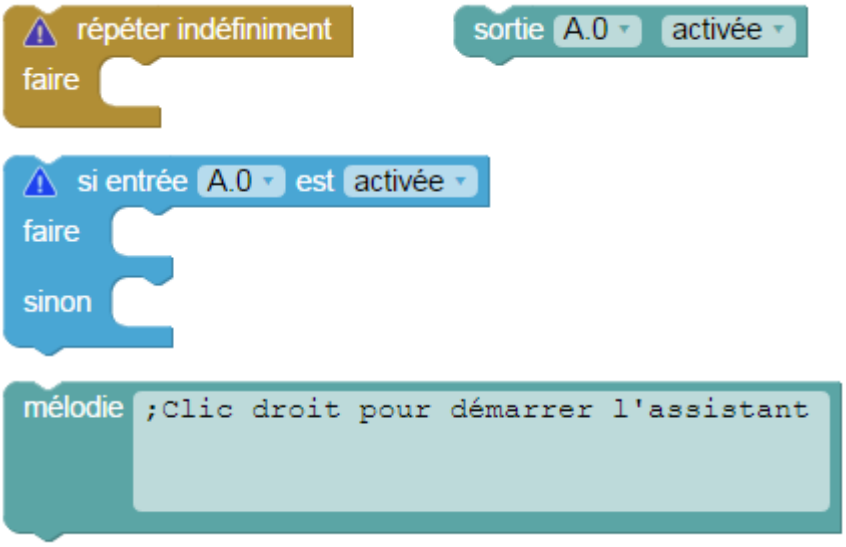
Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_B4_Organigramme.plf	Fichier Blockly : AL_N1_B4.xml

Remarque : Vous pouvez utiliser plusieurs boucles Si, nous verrons par la suite qu'il existe un moyen plus simple pour ne pas surcharger visuellement le programme.

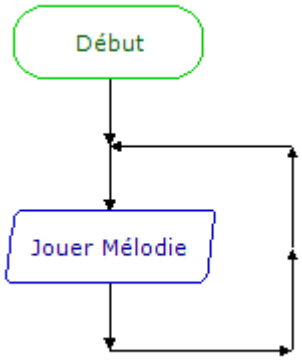
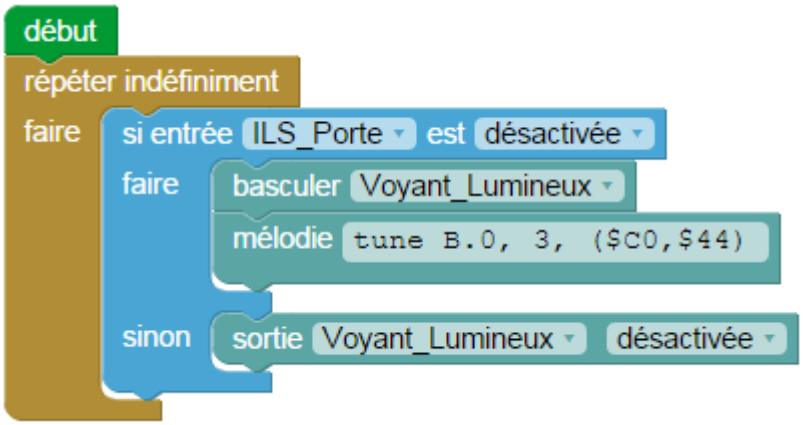
Exercice niveau 1 - B.5 : Créer votre propre alarme

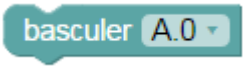
Objectif : Reprendre l'exercice précédent et rajouter les capteurs situés sur les fenêtres

Instructions utilisées :



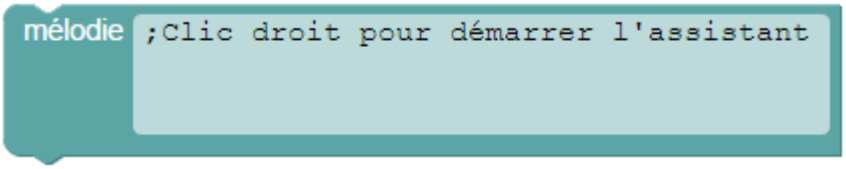
Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_B5_Organigramme.plf	Fichier Blockly : AL_N1_B5.xml

Remarque : La fonction 

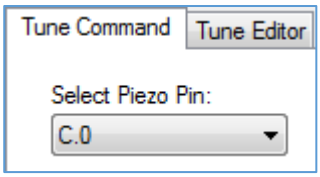
permet de changer le niveau du voyant lumineux à chaque passage dans la boucle.
Dans ce programme, elle allumera ou éteindra le voyant lumineux à chaque fois que la mélodie est effectuée.

Instructions :



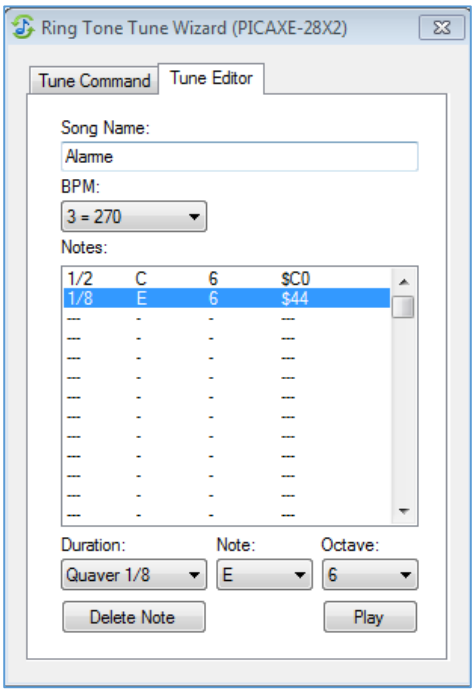
Clic droit sur l'assistant → Démarrer l'assistant

Choisir la sortie correspondant au buzzer dans la liste **Select Piezo Pin.**

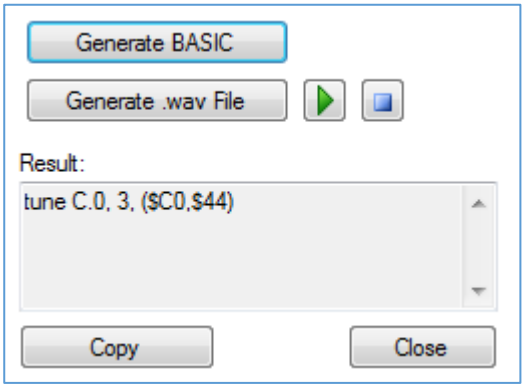


Choisir des notes pour créer l'alarme personnalisée à partir de l'onglet **Tune Editor**.

Exemple d'alarme classique :



Retourner dans l'onglet **Tune Command** et cliquer sur **Generate BASIC**



Recopier le résultat dans le bloc mélodie

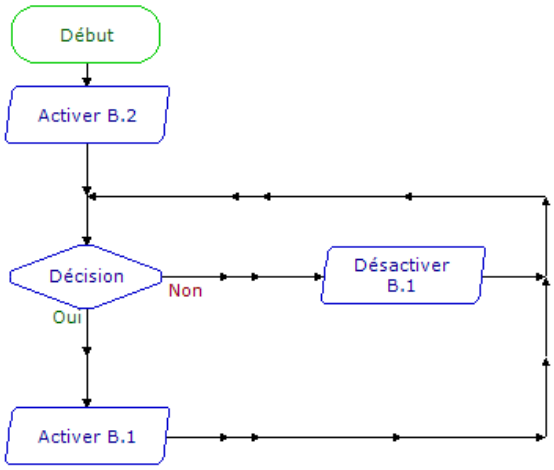
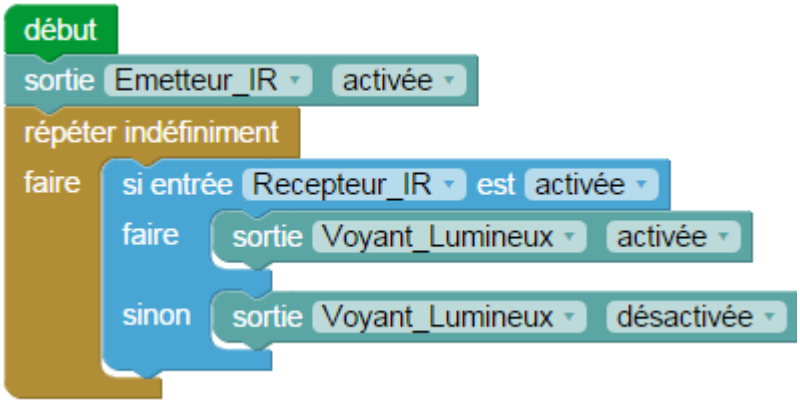
Niveau 1 - C

Exercice niveau 1 - C.1 : Utilisation d'un système émetteur/ récepteur infrarouge

Objectif : Allumer le voyant lumineux lorsqu'il y a un obstacle entre le récepteur et l'émetteur infrarouge

Instructions utilisées :

Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_C1_Organigramme.plf	Fichier Blockly : AL_N1_C1.xml

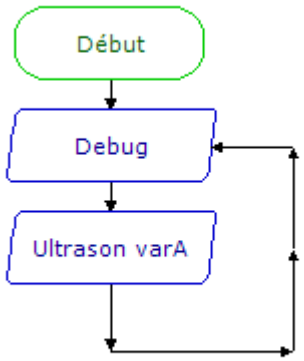
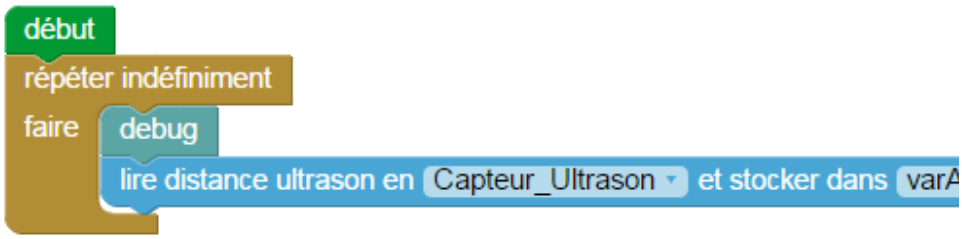
Remarque : Le récepteur est activé lorsqu'il n'y a rien, c'est à dire qu'il est désactivé lorsqu'il est en face d'un émetteur.

Exercice niveau 1 - C.2 : Utilisation d'un capteur à ultrason

Objectif : Récupérer la valeur de distance envoyée par un capteur à ultrason

Instructions utilisées :

Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_C2_Organigramme.plf	Fichier Blockly : AL_N1_C2.xml

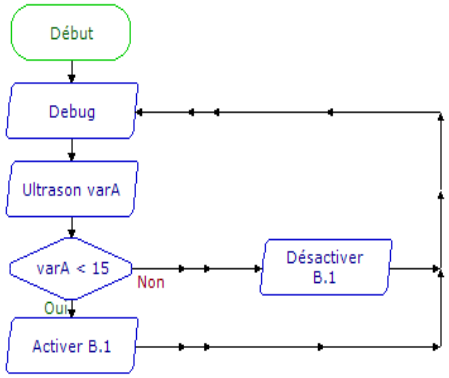
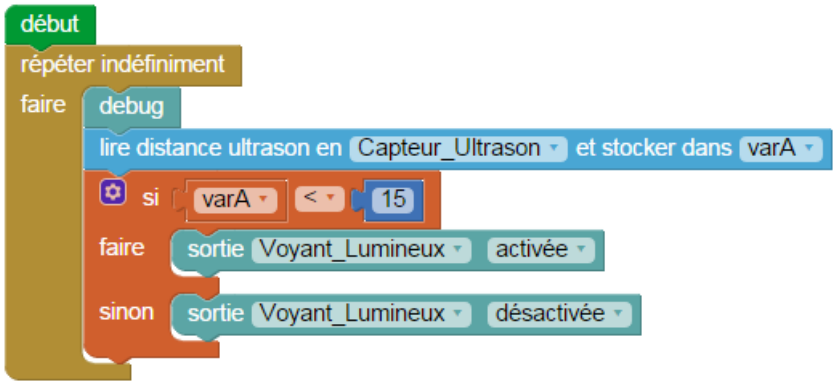
Remarque : Le capteur fonctionne de telle sorte qu'il donne la distance de l'objet sur lequel il pointe

Exercice niveau 1 - C.3 : Utilisation d'un capteur à ultrason

Objectif : Allumer le voyant lumineux lorsqu'il y a un obstacle trop proche du capteur à ultrason

Instructions utilisées :

Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_C3_Organigramme.plf	Fichier Blockly : AL_N1_C3.xml

Module capteur PIR

Le module PIR est équipé d'un capteur pyroélectrique. Il réagit aux faibles variations de température et permet de détecter la présence (mouvement) d'une personne jusqu'à 5 m. Son champ de détection et de 60° jusqu'à 2,5 m et 20° à 5 m.

Le capteur réagit comme un bouton poussoir actif lors d'une détection d'un mouvement. Son activation est retardée d'environ 20 secondes après la mise sous tension afin d'éviter les détections intempestives.

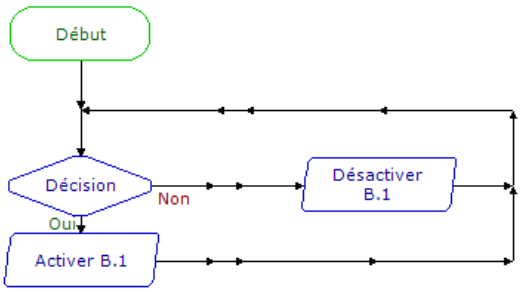
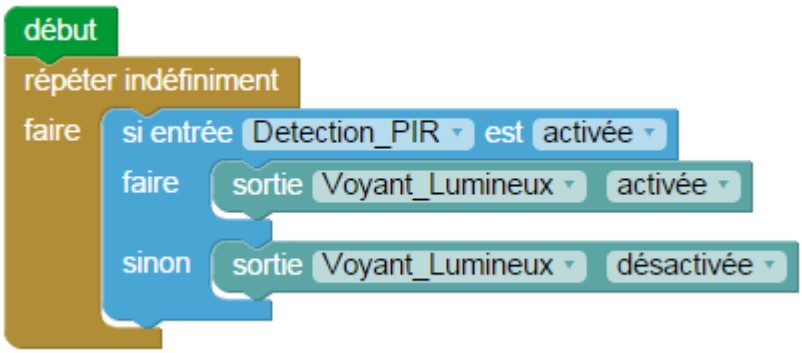
Par ailleurs, le capteur est sensible aux variations de températures brutales, aux vibrations ou aux chocs importants. Il ne faut pas l'exposer à la lumière directe du soleil, à l'air pulsé d'un radiateur ou d'un climatiseur. Il est conçu pour une utilisation en intérieur ; pour une utilisation en extérieur, une protection anti humidité est nécessaire.

Exercice niveau 1 - C.4 : Utilisation d'un capteur PIR

Objectif : Allumer le voyant lumineux lorsqu'il y a une détection sur le capteur PIR

Instructions utilisées :

Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : AL_N1_C4_Organigramme.plf	Fichier Blockly : AL_N1_C4.xml

Remarque : Le récepteur est activé lorsqu'il n'y a rien, c'est à dire qu'il est désactivé lorsqu'il est en face d'un émetteur.

Programmation version de base niveau 2

Objectifs :

- Utilisation concrète d'AutoAlarme
- Utilisation de tous les modules de la maquette.
- Appréhension des différentes fonctionnalités du matériel ainsi que certaines notions de sécurité.

Ce niveau permet de mettre en œuvre la maquette, au fur et à mesures des exercices vous allez utiliser de plus en plus de modules et enrichir votre code pour obtenir à la fin du niveau une maquette qui marche parfaitement et qui respecte une logique de fonctionnement calquée sur le réel.

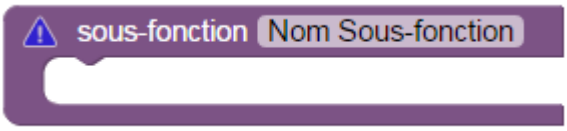
Nom du fichier	**Description**	**Objectif**
Niveau 2 A **Fichier modèle :** LU_N2_A.xml		
LU_N2_A1	Utilisation des sous-fonctions	Notions de programmation abordées : -Utilisation des sous-fonctions
LU_N2_A2	Personne ne bouge	

Niveau 2 - A

Exercice niveau 2 - A.1 : Utilisation des sous-fonctions

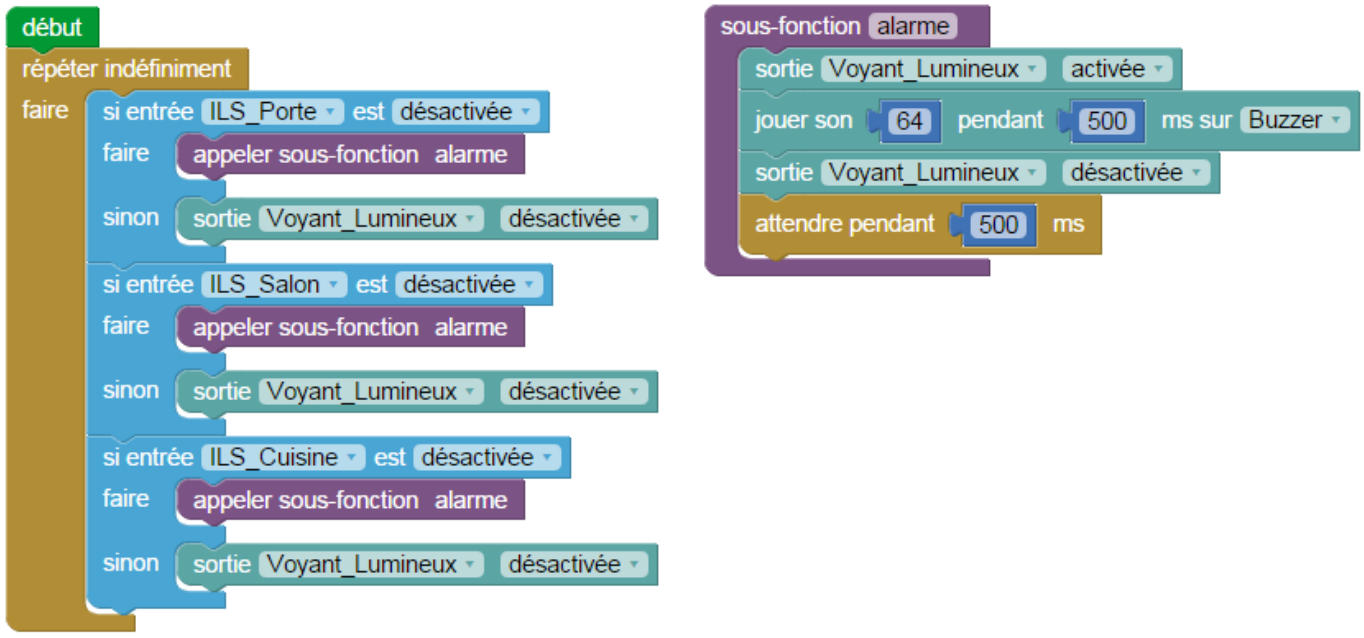
Objectif : Reprendre l'exercice AL_N1_B4.xml, remettre le programme d'alarme dans une sous-fonction qu'on devra appeler

Instructions utilisées :



Correction :

Blocs



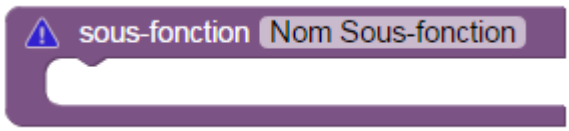
Fichier Blockly : AL_N2_A1.xml

Remarque : On peut appeler des sous-fonctions dans une sous-fonction

Exercice niveau 2 - A.2 : Personne ne bouge

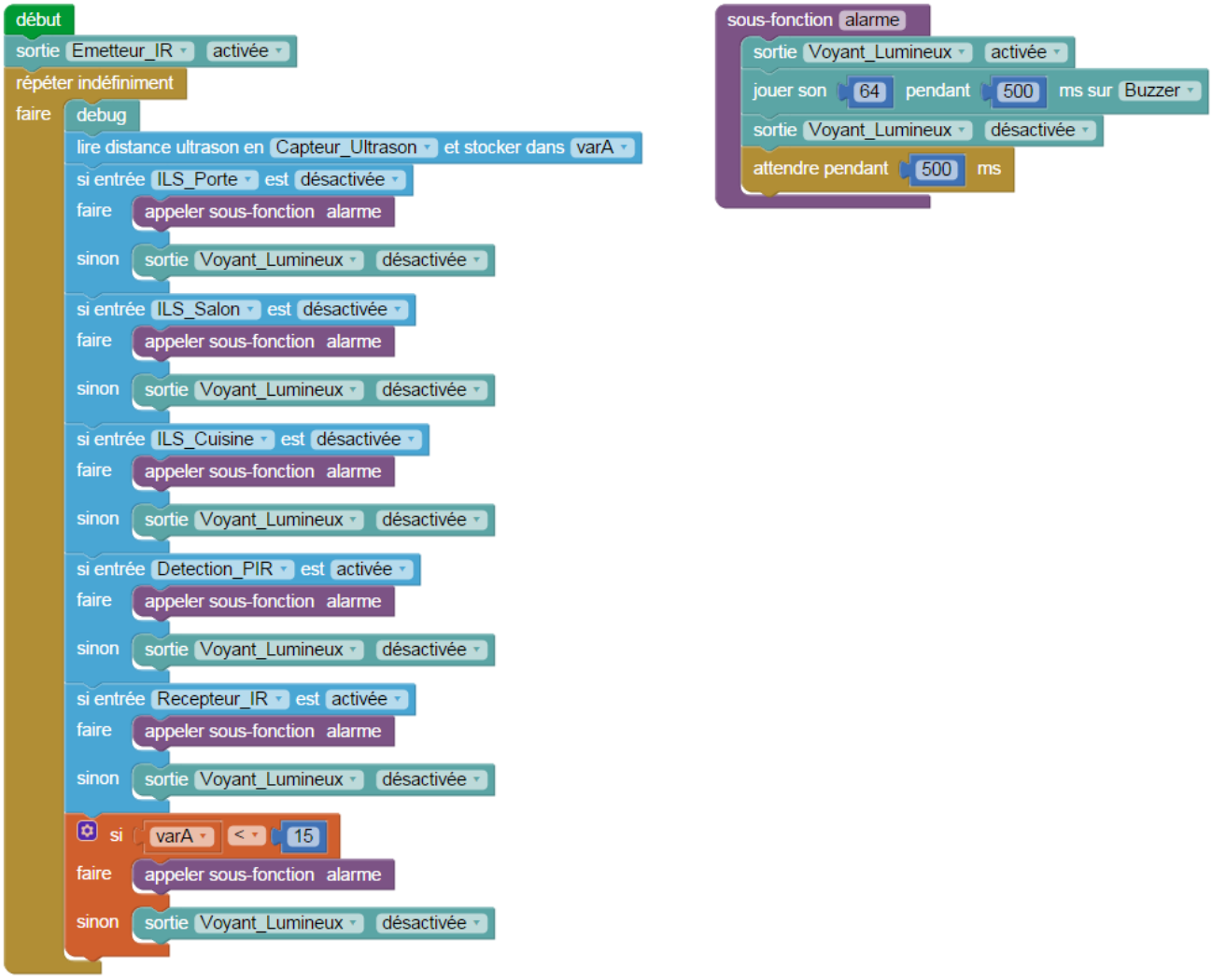
Objectif : Créer un programme qui active l'alarme lorsqu'il y a intrusion par une porte ou une fenêtre, ou lorsqu'un des capteurs est activé.

Instructions utilisées :



Correction :

Blocs



Fichier Blockly : AL_N2_A2.xml

Programmation niveau 3

Objectif :

- Utiliser les modules plus complexes : pilotage à distance, contrôle par le courant…

Le niveau 3 n'intègre pas de nouvelles notions de programmation mais de nouveaux blocs permettant d'utiliser les modules options.

Nom du fichier	Description	Objectif
Niveau 3 A – Module Bluetooth **Fichier modèle :** AL_N3_A.xml		
AL_N3_A1.xml	Contrôler l'alarme à l'aide de 2 boutons présent sur l'application Android.	Fonctionnalité matérielle abordé : - Module Bluetooth Notions de programmation abordées : - Liaison série (hserin/hserout).
AL_N3_A2.xml	Désactiver une alarme à l'aide d'un bouton présent sur l'application.	
AL_N3_A3.xml	Permet au programme de rendre l'alarme automatique ou non et de la couper en cas d'intrusion.	
Niveau 3 B – Télécommande infrarouge **Fichier modèle :** AL_N1_B.xml		
AL_N3_B1.xml	Allumer ou éteindre le voyant lumineux à l'aide de la télécommande IR.	Fonctionnalité matérielle abordé : Utilisation de la télécommande IR Notions de programmation abordées : Utilisation d'un block dédié à la communication IR.
AL_N3_B2.xml	Pouvoir activer ou désactiver l'alarme grâce à un code envoyé par la télécommande.	

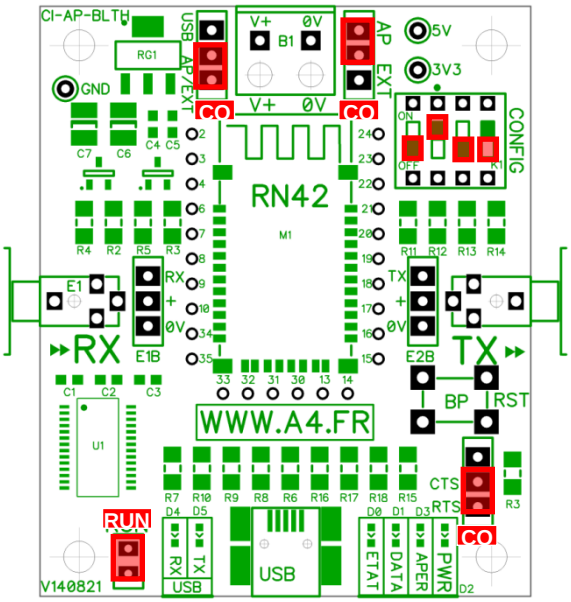
Option : Module Bluetooth

Le module Bluetooth développé par A4 Technologie permet de convertir le protocole Bluetooth en protocole de communication type Série qui est le mode de communication classique utilisé avec PICAXE ou Arduino. Ce module accepte différentes configurations.
En mode avancé, il peut être configuré au travers d'une liaison par connexion USB à un PC ou par l'envoi de commandes au travers de ses liaisons RX et TX.

La documentation technique du module Bluetooth décrit en détail les fonctionnalités du module. Elle est téléchargeable sur http://a4.fr/wiki/index.php/Module_Bluetooth_-_K-AP-MBLTH_/_S-113020008 .
Les informations seront envoyées via un smartphone ou une tablette possédant la technologie Bluetooth à l'aide d'une application développée sous AppInventor par l'équipe technique de A4.

Configuration

Positionner les cavaliers et interrupteurs comme indiqué par les positions repérées en rouge ci-dessous.



Le cavalier repéré **RUN** est utilisé lors de la mise au point de programmes avec **Arduino**. Il doit être ôté pour permettre le téléversement du programme puis doit être remis lors de l'utilisation.
La mise au point de programmes avec **PICAXE** ne nécessite pas d'ôter ce cavalier pour transférer le programme.

Les cavaliers **CO1** et **CO2** permettent de sélectionner le mode d'alimentation du module Bluetooth. Dans la configuration ci-dessus, son alimentation provient directement de l'interface AutoProg ou AutoProgUno au travers des cordons de liaison avec le module ; ils sont positionnés respectivement sur AP et sur AP/EXT.

Le cavalier **CO3** est utilisé en mode avancé pour relier ou dissocier les signaux CTS et RTS nécessaires au fonctionnement du module Bluetooth. Ici, il est positionné sur CTS/RTS.

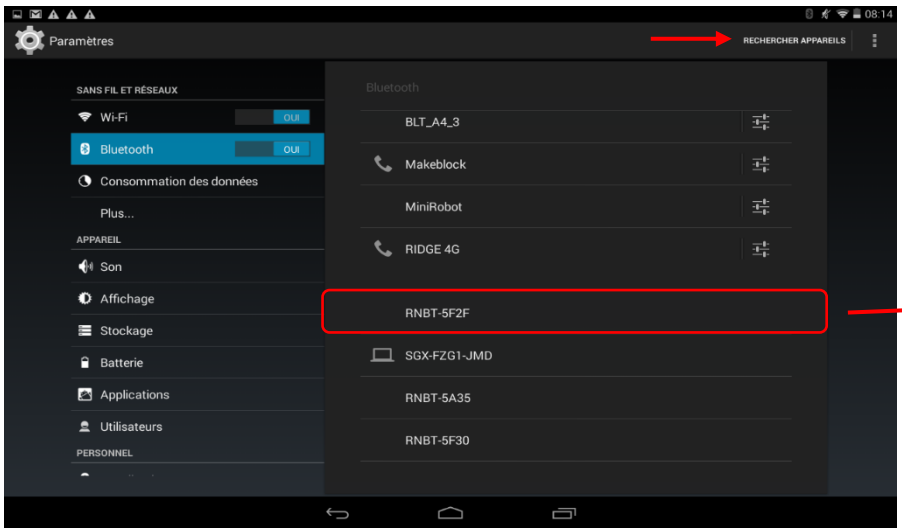
Les interrupteurs **CONFIG** permettent de paramétrer le mode de fonctionnement du module Bluetooth. Ici, l'interrupteur n°2 est positionné sur ON pour sélectionner une vitesse de transmission des données à 9600 bauds.

Témoins lumineux

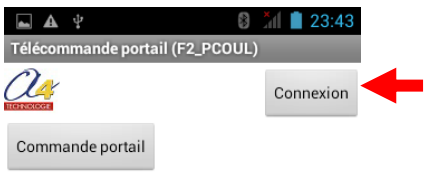
PWR indique que le module est sous tension.
APER indique que le module est associé avec un matériel Bluetooth.
DATA indique qu'il y a un flux de données entre le module et l'appareil avec lequel il est connecté.
ETAT indique que le module est opérationnel. L'affichage clignotant indique qu'il n'est pas opérationnel.
USB RX indique qu'il y a un flux de données sur la liaison USB du PC vers le module.
USB TX indique qu'il y a un flux de données sur la liaison USB du module vers le PC.

Mise en place des programmes et procédure de connexion

Avant de commencer à tester les programmes il faut d'abord appairer le smartphone ou la tablette au module bluetooth.
Pour cela rendez-vous dans les réglages bluetooth et lancer une recherche d'appareils (la maquette doit être allumée pour allimenter le module). Le nom de votre module s'appelle : RNBT + les 4 derniers chiffres de l'adresse mac du module notés sur le composant. Selectionnez le et un message proposant de vous connecter à lui deARait s'afficher.



Une fois cette étape passée vous pourrez vous connecter au module à partir du programme AppInventor à chaque fois.
Lorsque la connexion est réalisée, le bouton **Déconnexion** apparaît dans l'application.
Le témoin vert **DATA** s'allume sur le module dès qu'une donnée est émise ou reçue par le module Bluetooth.
L'appui sur le bouton d'envoi de données, dans cet exemple **Commande portail**, déclenche l'allumage fugitif de ce témoin.



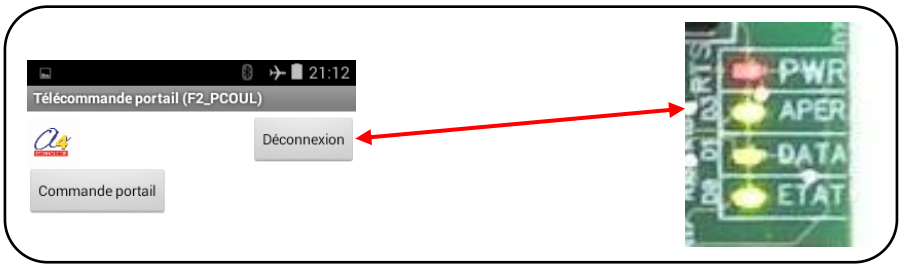
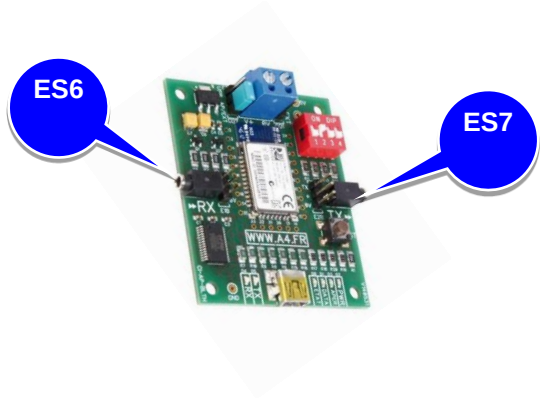


Tableau d'affectation des entrées et sorties en Bluetooth

	Modules de communication pour entrées / sorties numériques	Broche Blockly	Etiquette Blockly
7	Communication Bluetooth envoi de données	C.7	**BLTH_TX***
6	Communication Bluetooth réception de données	C.6	**BLTH_RX***
EN	**Modules capteurs pour entrées numériques**		
5	Capteur à ultrasons	C.5	Capteur_Ultrason
4	Récepteur infrarouge	*C.4*	Recepteur_IR
3	Capteur PIR	C.3	Detection_PIR
2	Capteur magnétique de la porte d'entrée	C.2	ILS_Porte
1	Capteur magnétique de la fenêtre du salon	C.1	ILS_Salon
0	Capteur magnétique de la fenêtre de la cuisine	C.0	ILS_Cuisine
EA	**Modules capteurs pour entrées analogiques**		
3	*(libre)*	*A.3*	
2	*(libre)*	A.2	
1	*(libre)*	A.1	
0	*(libre)*	A.0	
SN	**Modules actionneurs sorties numériques**		
7	*(libre)*	B.7	
6	*(libre)*	B.6	
5	*(libre)*	B.5	
4	*(libre)*	B.4	
3	*(libre)*	B.3	
2	Emetteur infrarouge	B.2	Emetteur_IR
1	Borne reliée au buzzer	B.1	Buzzer
0	Module signal LED jaune	B.0	Voyant_Lumineux

Câblage du module Bluetooth (K-AP-MBLTH)



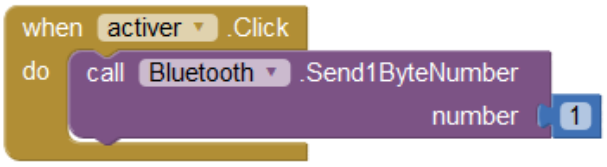
Exercice niveau 3 - A.1 : Ouvrir/fermer avec application Bluetooth

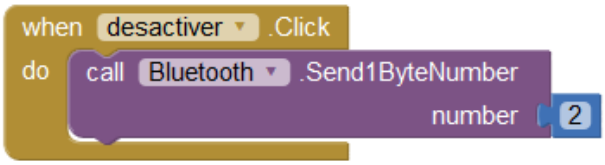
Objectif : contrôler l'alarme via Bluetooth à partir d'une application pour Smartphone

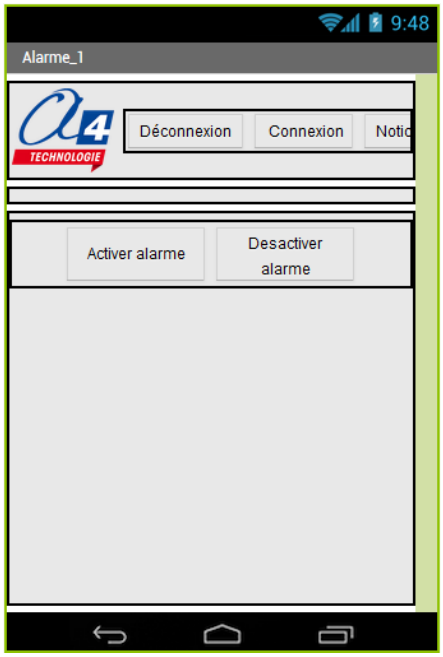
Notion abordée : réception de données Bluetooth envoyées par un Smartphone.

Application Android : Alarme_1.apk

Fichier App Inventor : Alarme_1.aia

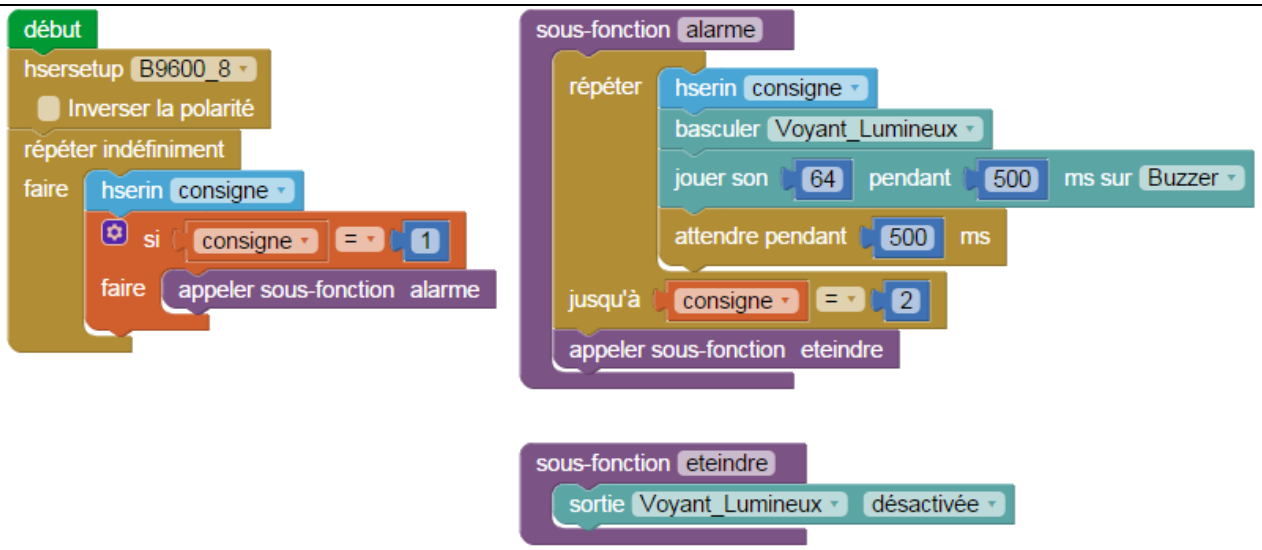






Correction :

Blocs



Fichier Blockly : AL_N3_A1.xml

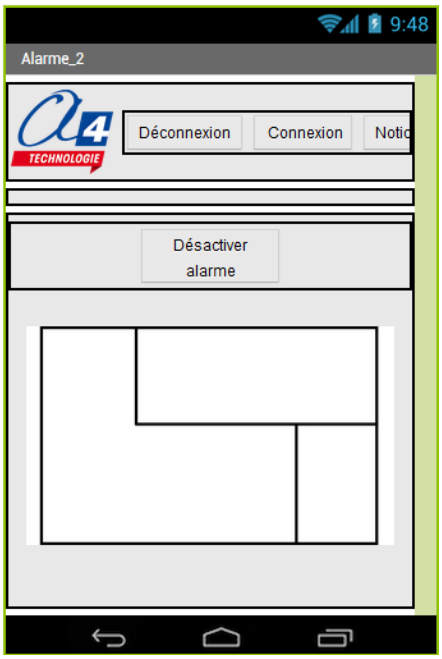
Exercice niveau 3 - A.2 : Détecter une intrusion via Bluetooth

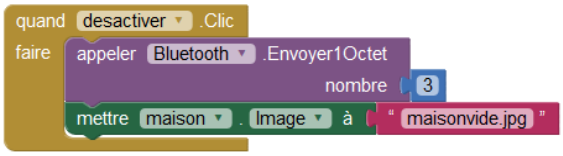
Objectif : Détecter où se situe l'intrusion ; Activer l'alarme cas d'intrusion, celle-ci pourra être désactivée grâce à un bouton sur l'application.

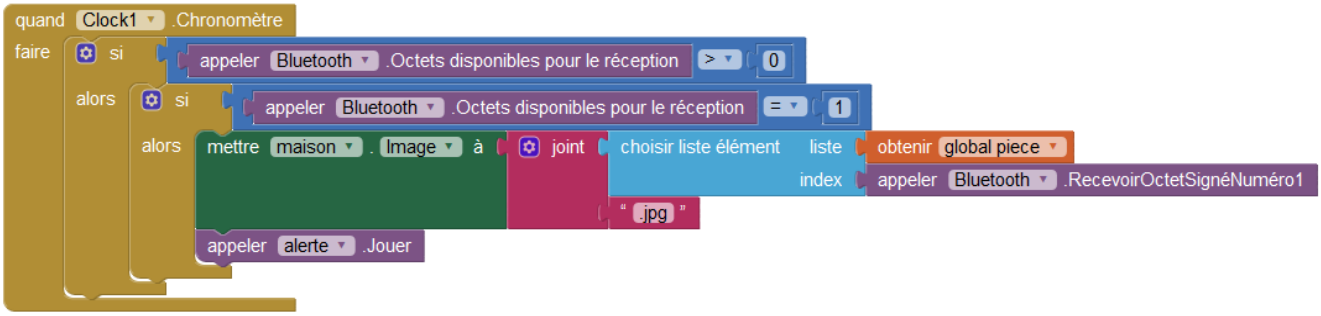
Notion abordée : réception de données Bluetooth envoyées par un Smartphone.

Application Android : Alarme_2.apk

Fichier App Inventor : Alarme_2.aia

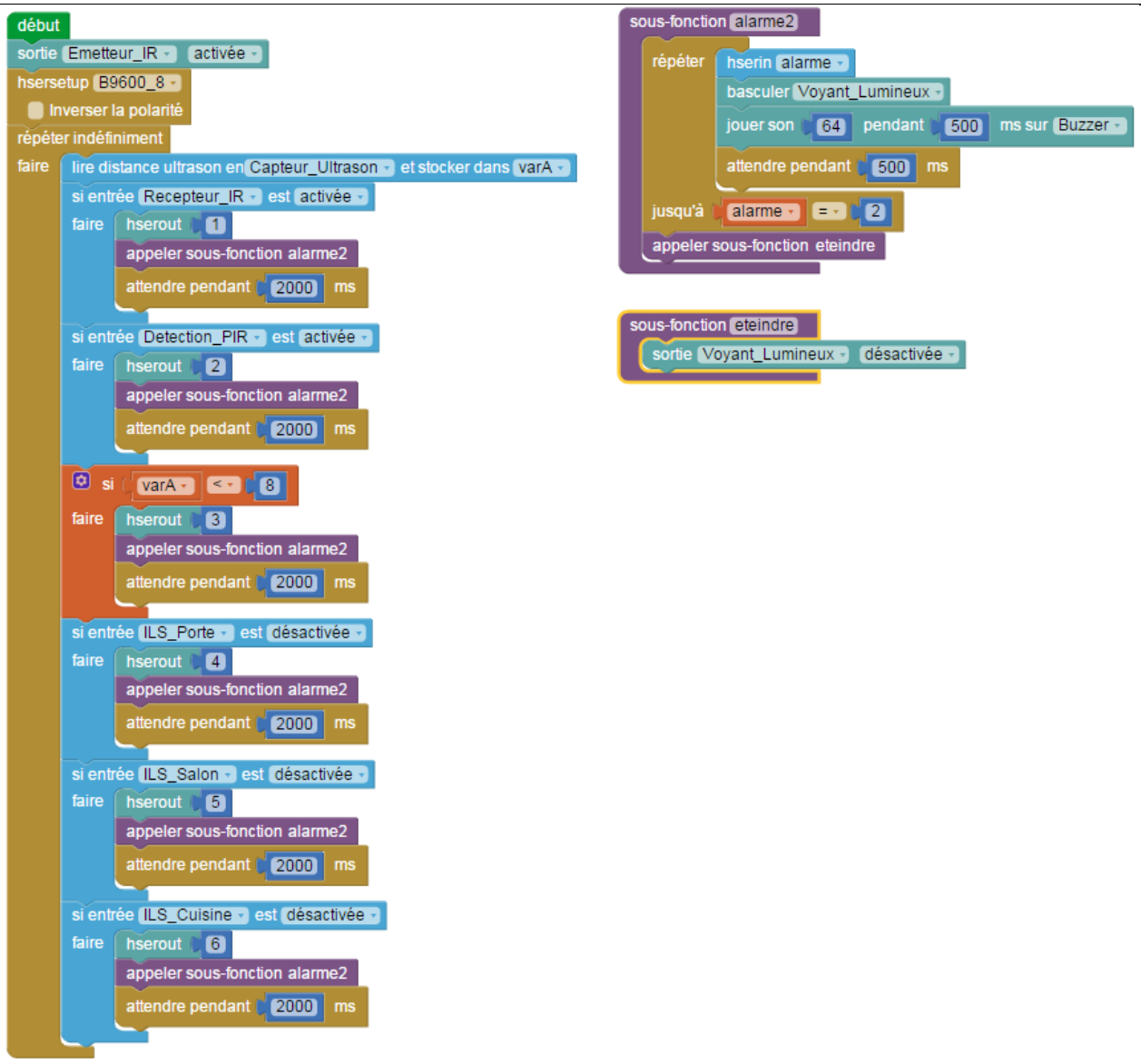






Correction :

Blocs



Fichier Blockly : AL_N3_A2.xml

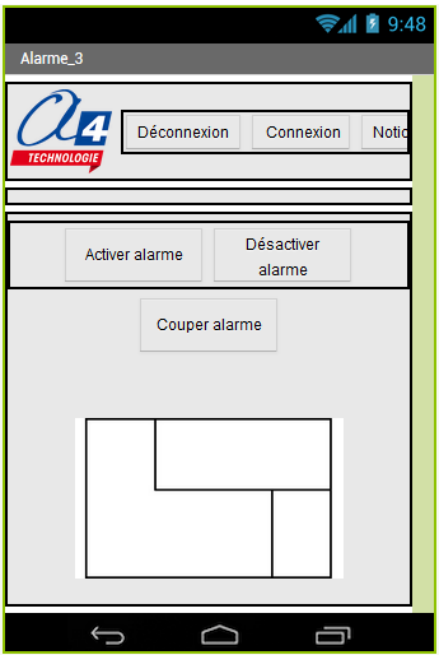
Exercice niveau 3 - A.3 (difficile): Système d'alarme activable

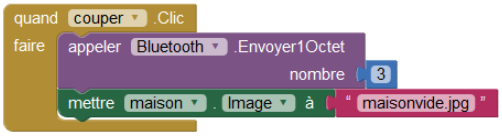
Objectif : Détecter où se situe l'intrusion ; Activer l'alarme ou non en cas d'intrusion

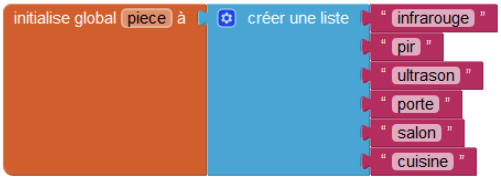
Notion abordée : Créer un programme permettant à l'alarme d'être activée ou désactivée. C'est-à-dire qu'il va ou non chercher s'il y a une intrusion.

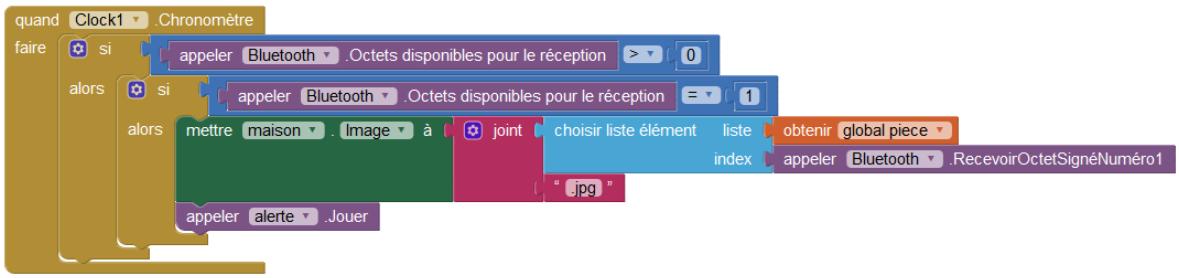
Application Android : Alarme_3.apk

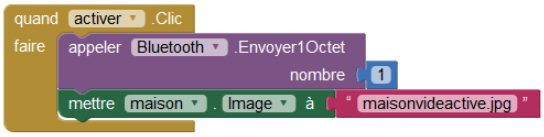
Fichier App Inventor : Alarme_3.aia

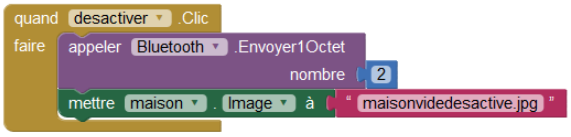








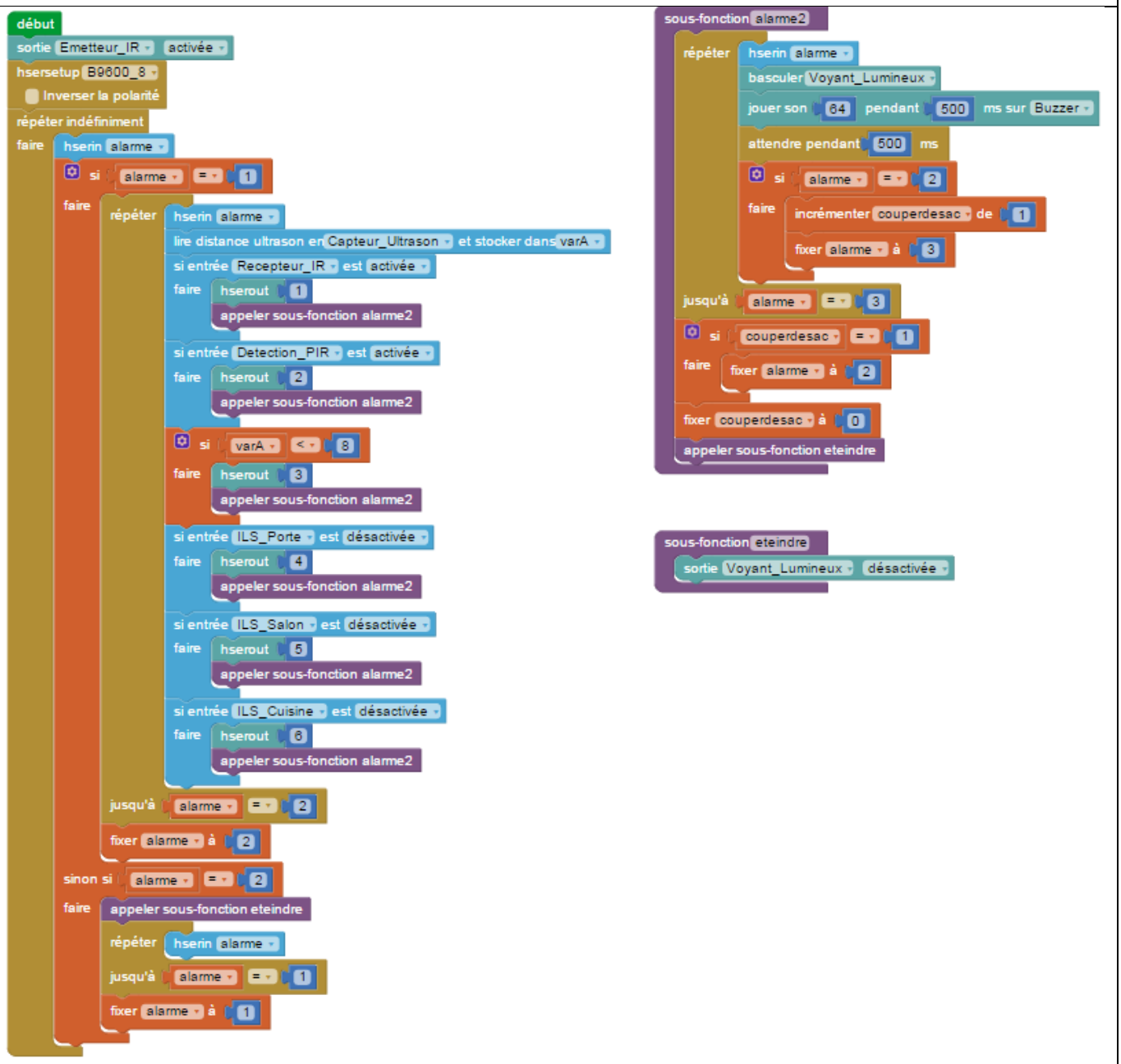




Remarque : Les différents jpg correspondent aux images jouées sur l'application.

Correction :

Blocs



Fichier Blockly : AL_N3_A3.xml

Remarques : La variable « couperdesac » correspond au fait qu'on arrête l'alarme également en appuyant sur « Désactiver l'alarme »

Le bouton « couper l'alarme » éteint simplement l'alarme, mais elle pourra se déclencher à nouveau.

Une astuce pour savoir si l'alarme est activée ou non est de regarder l'état de la LED derrière les capteurs à Ultrasons. Si elle est allumée, c'est que le capteur fonctionne et donc que l'alarme est active.
Si on appuie deux fois sur désactiver, il faudra appuyer deux fois sur activer, même si l'application donne pour indication que l'alarme est activée lors d'un premier appui.

Option : Module télécommande infrarouge

Télécommande RAX-TVR10

La télécommande universelle infrarouge associée à un capteur infrarouge approprié permet de piloter à distance une carte PICAXE.
Afin d'assurer la compatibilité de fonctionnement avec le système PICAXE il est nécessaire de l'initialiser avec le mode de fonctionnement au standard *« Sony TV ».*

Attention : L'émetteur infrarouge doit toujours être désactivé lors de l'utilisation de la télécommande infrarouge.

Procédure de mise en service pour PICAXE

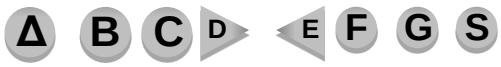
1. Insérer 2 piles AAA dans le logement au dos de la télécommande.

2. Appuyer simultanément sur S et B.
= La LED s'allume.

3. Taper le code 0 - 1 - 3
= La LED clignote brièvement à chaque appui des touches « 0 » et « 1 » puis s'éteint après l'appui sur la touche « 3 ».

4. Appuyer sur le bouton de mise en service. 
= La télécommande est opérationnelle.

Les touches suivantes risquent de déprogrammer :

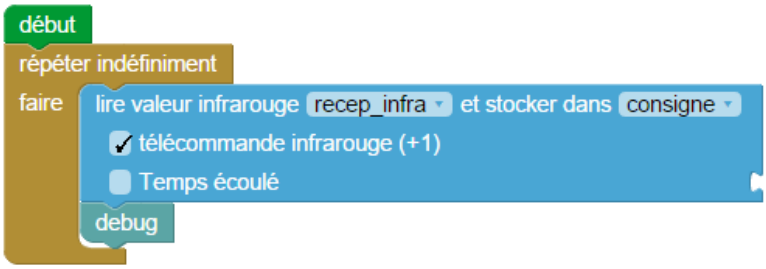


Conseil : Si la télécommande ne fonctionne plus, appuyer sur  pour revenir à la configuration compatible PICAXE

Remarque : Le guide d'utilisation complet de la télécommande est disponible ici :
http://www.a4telechargement.fr/RAX-TVR010/RAX-TRV10_Telecommande_InfraRouge.pdf

Tester la télécommande

Charger les programmes de test de la télécommande : « test_infra_bloc.xml » ou « test_infra_org.plf ».
Respecter le plan de câblage vu précédemment dans le dossier.



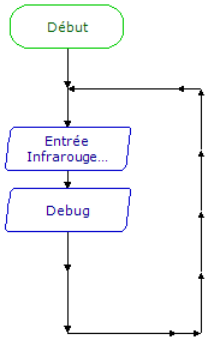


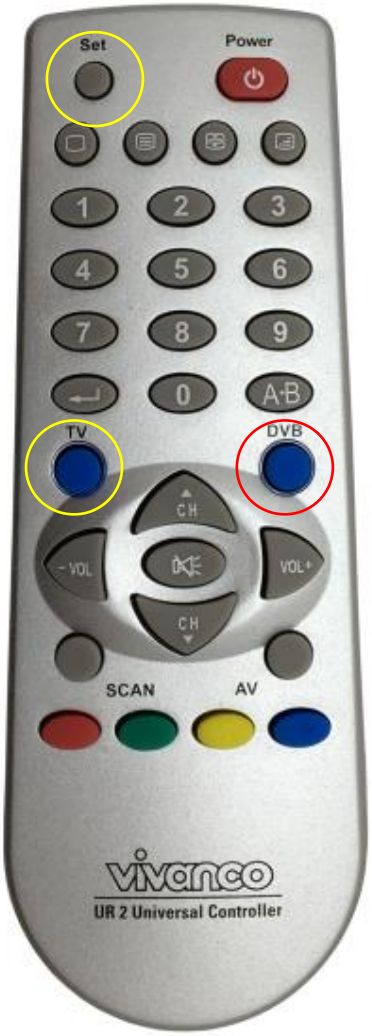
Tableau de correspondance des touches

Diriger la télécommande vers le récepteur infrarouge et vérifier dans la partie « Variables » de Picaxe Editor que les données reçues sont correctes.
Ci-dessous, le tableau des valeurs renvoyées par les différents boutons de la télécommande :

Touche	1	2	3	4	5	6	7	8	9	0	
Code émis	0	1	2	3	4	5	6	7	8	9	21
Code émis	16	17	19	18	96	54	37	20	98	11	

Télécommande TELEC-IR-UNIV

Procédure de mise en service pour PICAXE



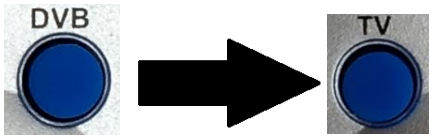
1. Insérer 2 piles AAA dans le logement au dos de la télécommande.

2. Appuyer simultanément sur les boutons **Set** et **TV**.
   Le bouton **Power** s'allume.

3. Taper le code 0-7-7.
   Le bouton **Power** clignote brièvement à chaque appui, puis s'éteint.

4. Appuyer sur le bouton **Power**.
   La télécommande est opérationnelle.

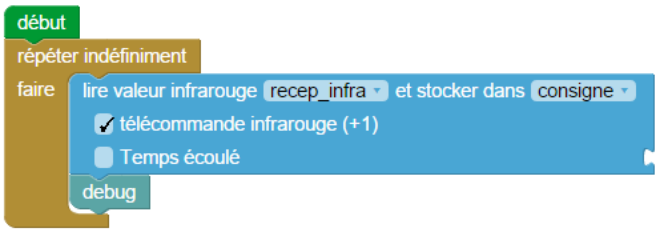
ATTENTION !
La touche **DVB** risque de changer le mode. Appuyer sur **TV** pour revenir dans le bon mode.



Conseil : Si la télécommande ne fonctionne plus, appuyer sur **TV** pour revenir à la configuration compatible PICAXE.

Tester la télécommande

Charger les programmes de test de la télécommande : « test_infra_bloc.xml » ou « test_infra_org.plf ».
Respecter le plan de câblage vu précédemment dans le dossier.



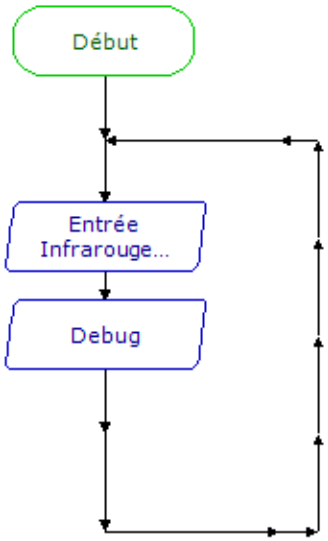
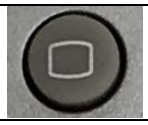
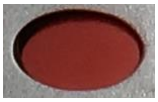


Tableau de correspondance des touches

L'appui sur une touche provoque l'émission d'un signal infrarouge qui véhicule un code correspondant à la touche. Par défaut : appui sur la touche 1 = envoi du code 0.

Pour simplifier l'utilisation de la télécommande infrarouge, il est possible d'activer la compatibilité entre le N° des touches et le code envoyé. Dans ce cas, appui sur la touche 1 = envoi du code 1.

Touche					
Code émis standard	**9**	**0**	**1**	**2**	**3**
Compatibilité activée	**10**	**1**	**2**	**3**	**4**
Touche					
Code émis standard	**4**	**5**	**6**	**7**	**8**
Compatibilité activée	**5**	**6**	**7**	**8**	**9**
Touche					
Code émis standard	**12**	**37**	**16**	**37**	**56**
Compatibilité activée	**13**	**38**	**17**	**38**	**57**
Touche					
Code émis standard	**63**	**74**	**29**	**21**	**76**
Compatibilité activée	**64**	**75**	**30**	**22**	**77**
Touche					
Code émis standard	**77**	**78**	**79**	**18**	**19**
Compatibilité activée	**78**	**79**	**80**	**19**	**20**
Touche					
Code émis standard	**20**	**16**	**17**		
Compatibilité activée	**21**	**17**	**18**		

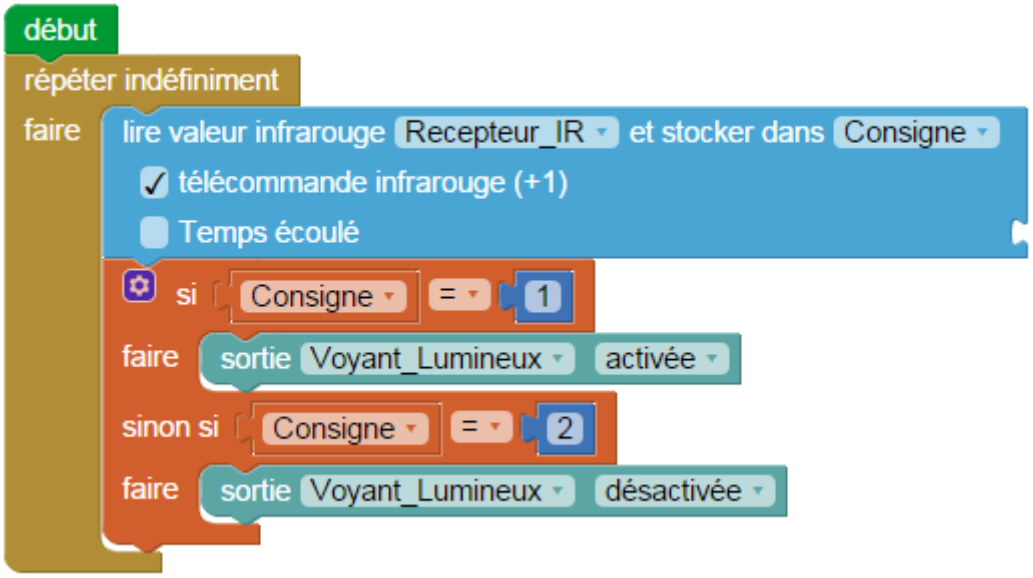
Exercice niveau 3 - B.1 : Contrôle voyant lumineux avec la télécommande IR

Objectif : Allumer ou éteindre le voyant lumineux à l'aide de la télécommande IR.

Notion abordée : gestion d'une liaison infra-rouge : télécommande/AutoProg à l'aide du bloc prévu à cet effet.

Correction :

Blocs



Fichier Blockly : AL_N3_A1.xml

Remarque : Cocher la case +1 permet d'avoir une concordance entre la touche pressée et la consigne reçue.

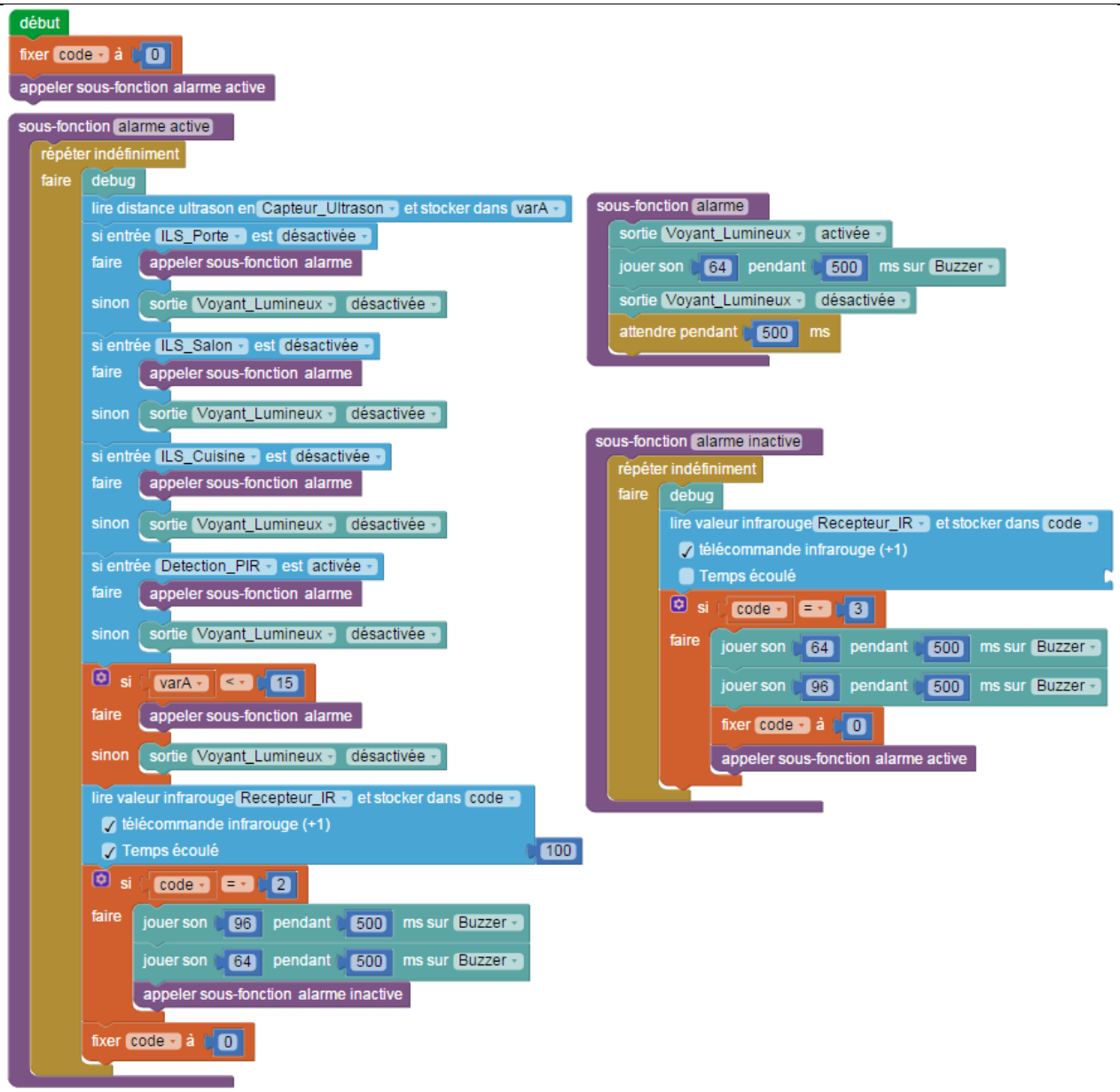
Exercice niveau 3 - B.2 : Activer / Désactiver l'alarme à l'aide de la télécommande IR

Objectif : Pouvoir activer ou désactiver l'alarme à partir de boutons sur la télécommande

Notion abordée : Utilisation du timeout sur la lecture infrarouge

Correction :

Blocs



Fichier Blockly : AL_N3_B2.xml

Remarque : Cocher la case timeout permet à la fonction de laisser place aux autres fonctions. Si on ne coche pas la case, la fonction va rester jusqu'à ce qu'on lui envoie une donnée et donc les autres fonctions ne seront pas atteignables.

Attention : Ne pas choisir un code envoyé à 1 avec un temps écoulé

CONCEPTEUR ET FABRICANT DE MATÉRIELS PÉDAGOGIQUES