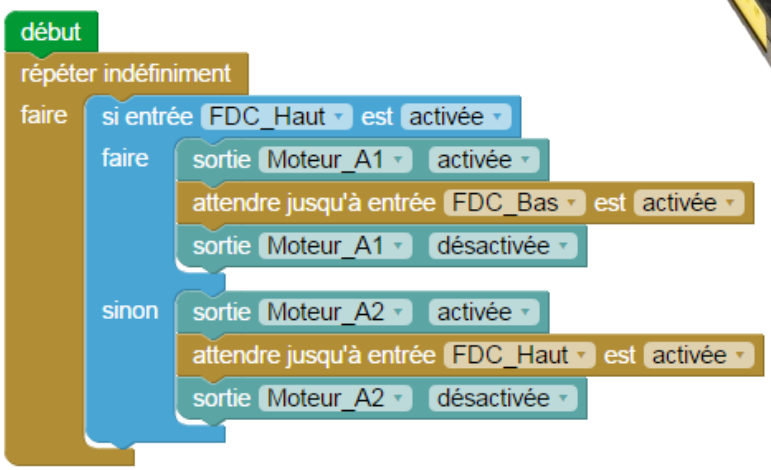
Plateforme élévatrice

Maquette programmable avec Editor / Blockly pour PICAXE



TECHNOLOGIE

Ressources disponibles pour le projet

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

Plateforme élévatrice

- Fichiers **3D** (SolidWorks, Edrawings et Parasolid) de la maquette et de ses options.
- Dossier **technique** pour la mise en œuvre de la maquette ;

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.

Les documents techniques et pédagogiques signés A4 Technologie sont diffusés librement sous licence Creative Commons BY-NC-SA :

– **BY** : Toujours citer A4 Technologie comme source (paternité).

– **NC** : Aucune utilisation commerciale ne peut être autorisée sans l'accord préalable de la société A4 Technologie.

– **SA** : La diffusion des documents éventuellement modifiés ou adaptés doit se faire sous le même régime.

Consulter le site ***http://creativecommons.fr/***

Note : la duplication de ce dossier est donc autorisée sans limite de quantité au sein des établissements scolaires, aux seules fins pédagogiques, à condition que soit cité le nom de l'éditeur A4 Technologie.

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

SOMMAIRE

Introduction

Plateforme élévatrice

La maquette Plateforme élévatrice (BE-AHANDI) est une reproduction homothétique d'une plateforme automatisée réelle : plusieurs étages, capteurs fin de course, moteur, plateforme, 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 de plateforme :

- 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 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 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 :

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

Exemple : PE_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 :** http://www.picaxe.com/Software
- **Maquette** Plateforme élévatrice (Réf. BE-AHANDI).
- **Câble de programmation** Picaxe USB (Réf : CABLE-USBPICAXE).
- **Interface programmable** AutoProgX1 ou X2 (Réf. K-APV2).
- C**ordons 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.

Environnement de programmation graphique

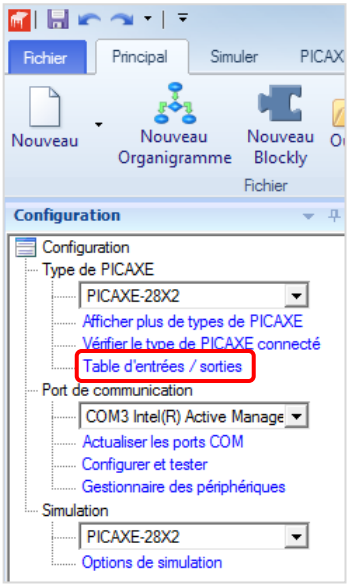
Tous les programmes correspondant aux activités ont été réalisés sous **PICAXE Editor 6**. En effet, ce logiciel de programmation graphique présente plusieurs avantages :

- 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.

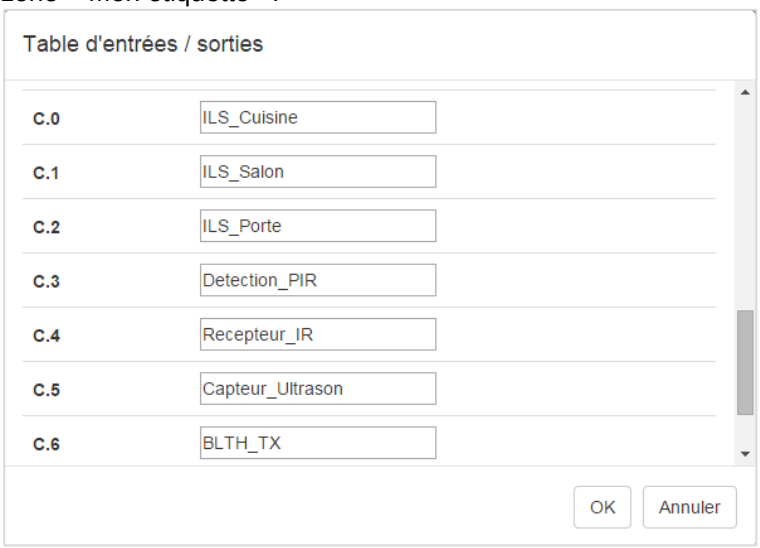
Note : vous pouvez aussi utiliser **Blockly for Picaxe** : environnement de programmation par blocs simplifié (nombre de menus limité et personnalisation des entrées/sorties non disponibles).

Personnalisation des entrées/ sorties

Nous vous proposons le fichier **PE_BASE.xml** dans lequel les noms des entrées/sorties ont été personnalisés pour une utilisation avec la maquette. Tous les programmes et activités proposés dans ce document se basent sur cette liste. Celle-ci reste modifiable à tout moment.
A partir de Picaxe Editor 6, dans l'explorateur d'espace de travail cliquer sur **Table d'entrées / sorties**.



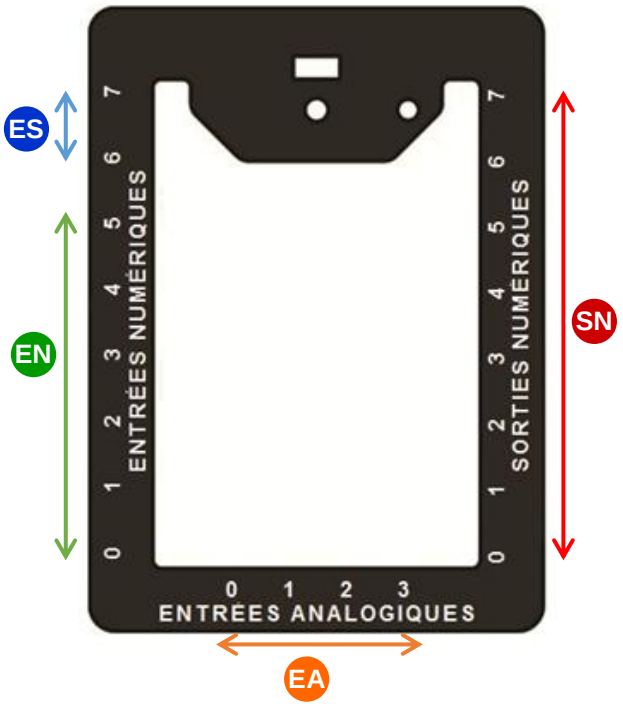
Une fenêtre apparaît à partir de laquelle vous pouvez modifier les noms de toutes les entrées et sorties dans la zone « Mon étiquette ».



Valider en cliquant sur **OK**.

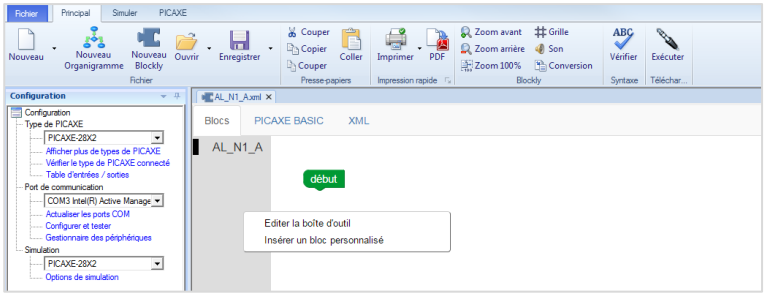
Tableau d'affectation des entrées et sorties

	Module 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	*(libre)*	C.5	
4	Bouton poussoir nacelle	C.4	BP_Nacelle
3	Bouton poussoir haut	C.3	BP_Haut
2	Capteur de fin de course de montée de la plate-forme	C.2	FDC_Haut
1	Capteur de fin de course de descente de la plate-forme	C.1	FDC_Bas
0	Bouton poussoir bas	C.0	BP_Bas
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	Connecté à la broche MOTA-2 de la carte contrôle moteur	B.7	Moteur_A2
6	Connecté à la broche MOTA-1 de la carte contrôle moteur	B.6	Moteur_A1
5	*(libre)*	B.5	
4	*(libre)*	B.4	
3	*(libre)*	B.3	
2	*(libre)*	B.2	
1	*(libre)*	B.1	
0	Module voyant lumineux	B.0	Voyant_Lumineux

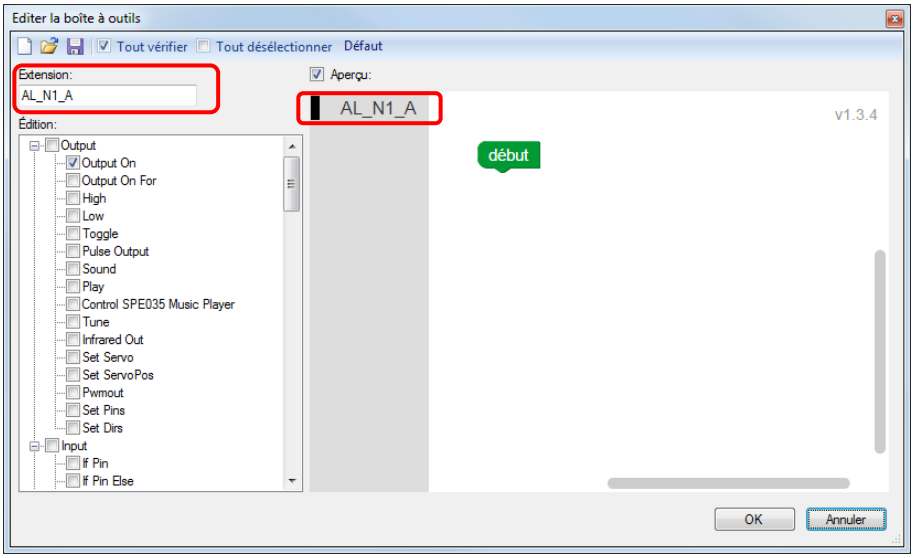


Personnalisation du jeu d'instructions

Vous pouvez personnaliser l'affichage du jeu d'instructions pour en limiter la quantité afin de faciliter la l'analyse et la correction des erreurs. Faire un clic droit sur la zone des blocs puis cliquer sur **Editer la boite d'outil**.



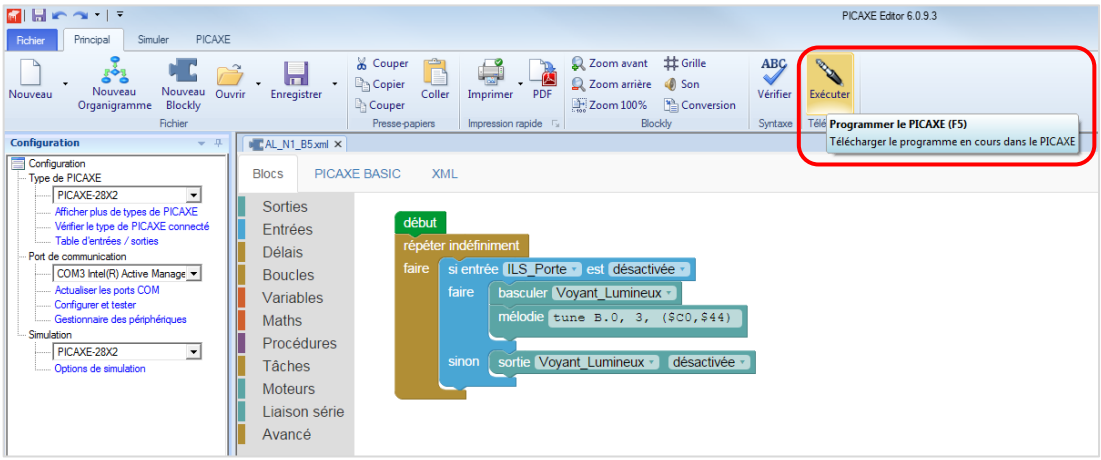
Une fenêtre s'ouvre à partir de laquelle vous pouvez sélectionner ou désélectionner les instructions de votre choix. Vous pouvez renommer le jeu d'instructions dans la zone « **Extension** ».



Valider en cliquant sur **OK**.

Procédure de chargement d'un programme

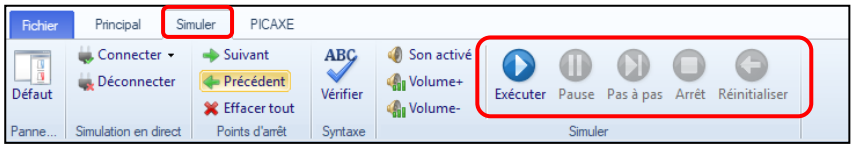
Commencer par relier le Loupiot à l'ordinateur avec le câble de programmation USB et le mettre sous tension.
A partir du Picaxe Editor 6, ouvrir un programme.



A partir du menu **Principal** ou du menu **PICAXE**, cliquer sur le bouton **Exécuter**. Vous pouvez également utiliser la touche **F5** de votre clavier.
Note : un programme téléchargé écrase le précédent.

Mode simulation

La simulation sur Picaxe EDITOR 6 permet de tester un programme avant de le téléverser dans la maquette.
Pour lancer et contrôler une simulation, utiliser les boutons **Exécuter / Pause / Pas à pas / Arrêt** à partir du menu **Simuler**.



La simulation surligne les blocs dans l'espace de travail pour vous montrer où en est le programme.

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 de la plateforme, 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 « PE_N1_A1.xml », charger le programme modèle « PE_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 :** PE_BASE.xml		
PE_N1_A1.xml	Activer un moteur dans un sens puis dans l'autre pour enfin s'arrêter.	Fonctionnalité matérielle abordée : -Gestion du moteur -Utilisation de Bouton-poussoir Notions de programmation abordées : -Boucle qui dépend d'une entrée
PE_N1_A2.xml	Ouvrir et fermer la plateforme en continu jusqu'à l'appui d'un bouton-poussoir.	
Niveau 1 B **Fichier modèle :** PE_BASE.xml		
PE_N1_B1.xml	Allumer un moteur à l'appui d'un bouton-poussoir.	Fonctionnalité matérielle abordée : -Utilisation de bouton-poussoir Notions de programmation abordées : -Le test d'une entrée (si/sinon)
PE_N1_B2.xml	Allumer les moteurs jusqu'à l'arrivée sur un capteur de fin de course puis repartir dans l'autre sens à chaque fin de course.	
PE_N1_B3.xml	Contrôler le moteur avec les boutons-poussoirs. Monter lors d'un appui sur le bouton du haut, descendre lors d'un appui sur le bouton du bas.	
PE_N1_B4.xml	Contrôler le moteur avec les boutons-poussoirs. Monter lors d'un appui sur le bouton du haut, descendre lors d'un appui sur le bouton du bas. Lors d'un appui sur le bouton de la nacelle, monter si la plate-forme était en bas et inversement.	
PE_N1_B1.xml	Allumer un moteur à l'appui d'un bouton-poussoir.	
Niveau 1 C **Fichier modèle :** PE_BASE.xml		
PE_N1_C1.xml	Incrémenter une variable au cours du temps et observer sa valeur à l'aide du PC (débogage).	Notions de programmation abordées : -Définition de variable -Incrémentation de variable -Test (si/sinon) de variable -Test (juste si) d'entrée -Débogage
PE_N1_C2.xml	Incrémenter une variable au cours du temps faire un test sur celle-ci pour activer un moteur.	
PE_N1_C3.xml	Incrémenter une variable à l'appui d'un bouton-poussoir, la décrémenter à l'appui de l'autre bouton-poussoir.	
PE_N1_C4.xml	Incrémenter une variable puis faire un test sur celle-ci pour contrôler un moteur.	

Niveau 1 - A

Exercice niveau 1 - A.1 : Maitriser la rotation du moteur.

Fichier modèle : PE_BASE.xml

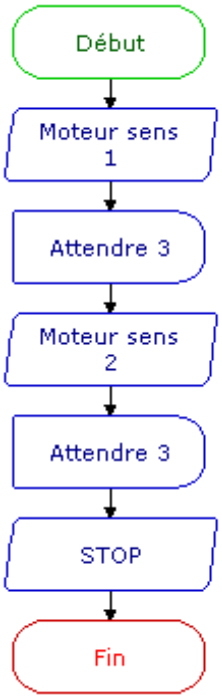
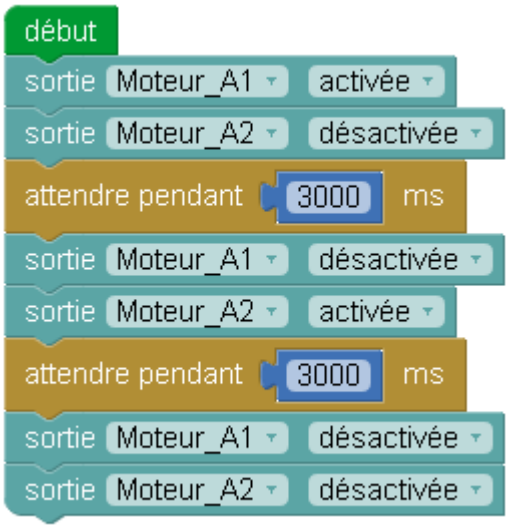
Objectif : activer un moteur dans un sens puis dans l'autre pour enfin s'arrêter.

Notion abordée : utilisation d'un moteur.

Remarque : Placer 2 charges sur les barres porte charges

Instructions utilisées :

Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_A1_Organigramme.plf	Fichier Blockly : PE_N1_A1.xml

ATTENTION : pour cet exercice il est recommandé de placer la plateforme à mi-hauteur pour éviter tout dommage. Il faut également activer le moteur à l'aide de l'interrupteur (Une LED rouge indique si le moteur est allumé).

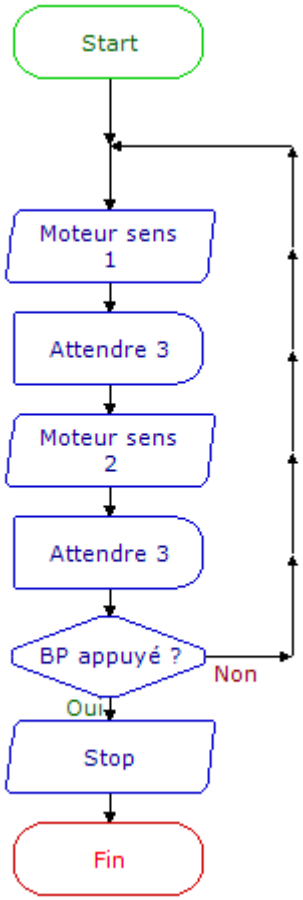
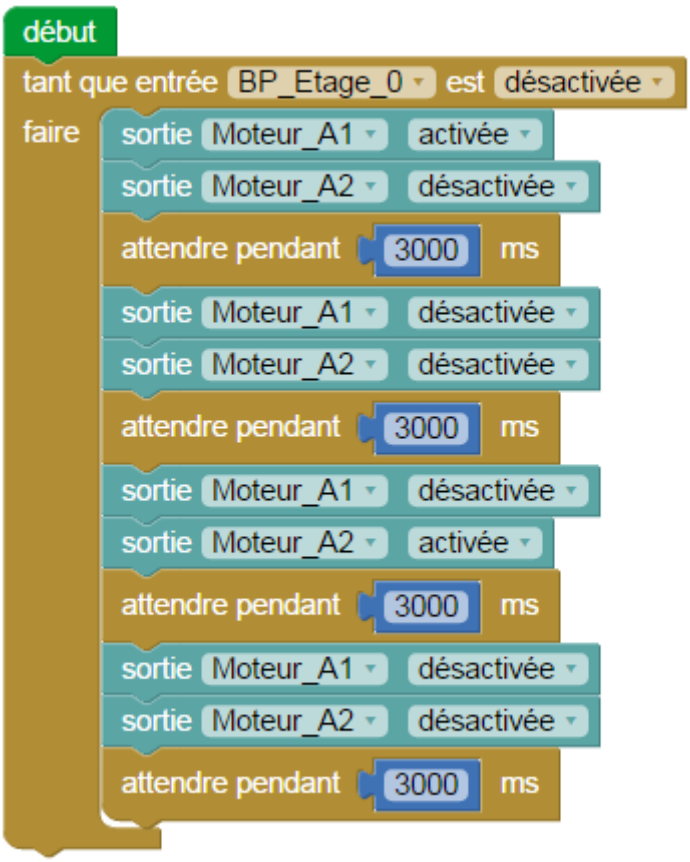
Exercice niveau 1 - A.2 : Utilisation d'une boucle tant que

Objectif : monter et descendre la plateforme en continu jusqu'à l'appui d'un bouton-poussoir.

Notion abordée : exécuter une boucle qui dépend de l'état d'une entrée.

Instructions utilisées :

Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_A2_Organigramme.plf	Fichier Blockly : PE_N1_A2.xml

Remarque : Le programme ne peut sortir de la boucle qu'une fois le test sur le bouton-poussoir validé. Le test sur le bouton poussoir se fait qu'une seule fois en début de séquence, avant de commencer l'ouverture. Si un appui est effectué pendant la séquence, aucun effet n'aura lieu sur le programme. Afin de vérifier à tout moment le changement d'état d'une entrée dans une séquence, l'utilisation des interruptions est indispensable (voir ex sur interruption).

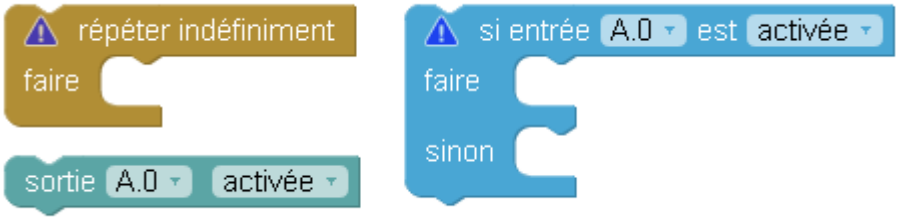
Niveau 1 - B

Exercice niveau 1 - B.1 : Instruction conditionnelle et bouton-poussoir

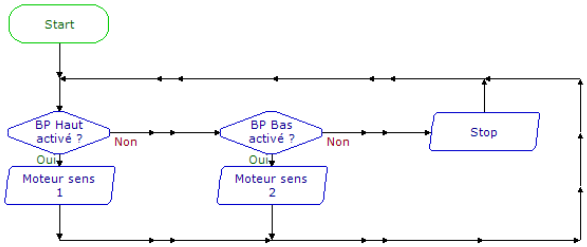
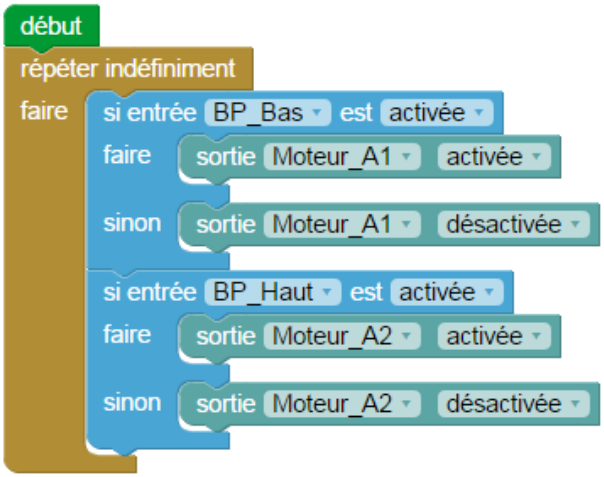
Objectif : allumer un moteur à l'appui d'un bouton poussoir

Notion abordée : utilisation des commandes conditionnelles (si/sinon).

Instructions utilisées :



Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_B1_Organigramme.plf	Fichier Blockly : PE_N1_B1.xml

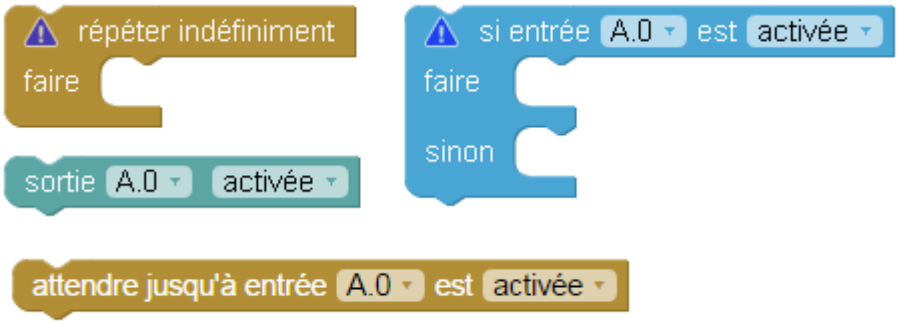
Remarque : les blocs de couleur bleu claires représente des commandes concernant l'utilisation des entrées.

Exercice niveau 1 - B.2 : Instruction conditionnelle, capteur de fin de course et délai

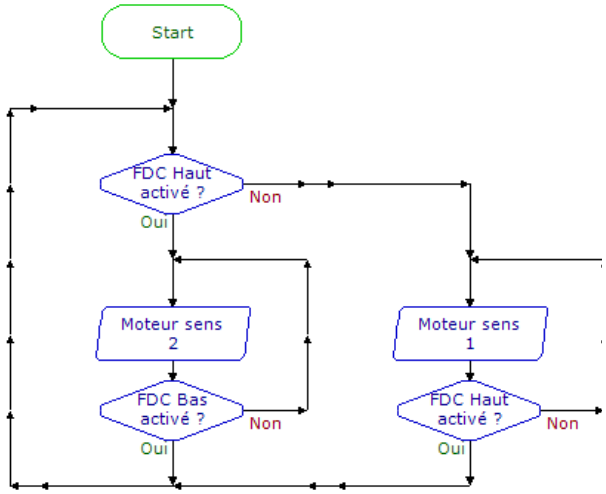
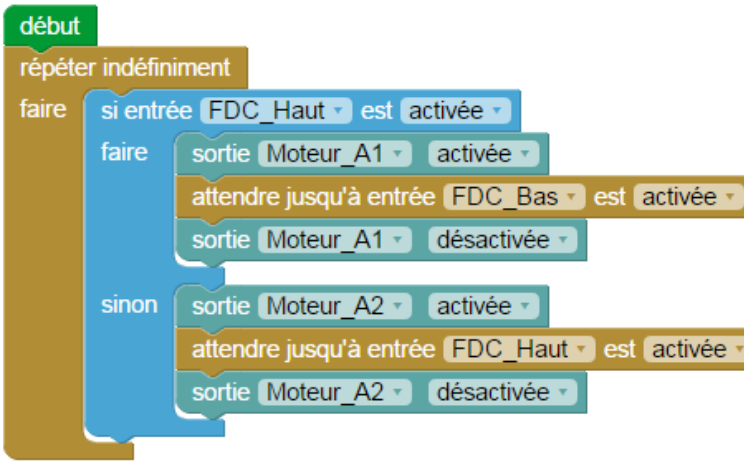
Objectif : allumer les moteurs jusqu'à l'arrivée sur un capteur de fin de course puis repartir dans l'autre sens à chaque fin de course

Notions abordées : utilisation des commandes conditionnelles (si/sinon) / utilisation d'un capteur fin de course / utilisation des délais.

Instructions utilisées :



Correction :

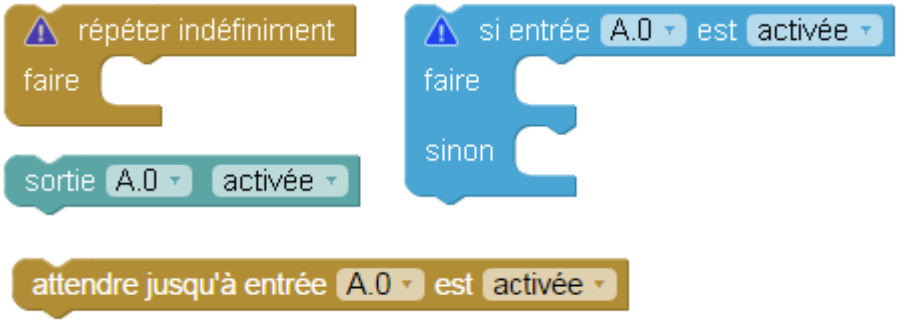
Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_B2_Organigramme.plf	Fichier Blockly : PE_N1_B2.xml

Exercice niveau 1 - B.3 : Contrôle moteur

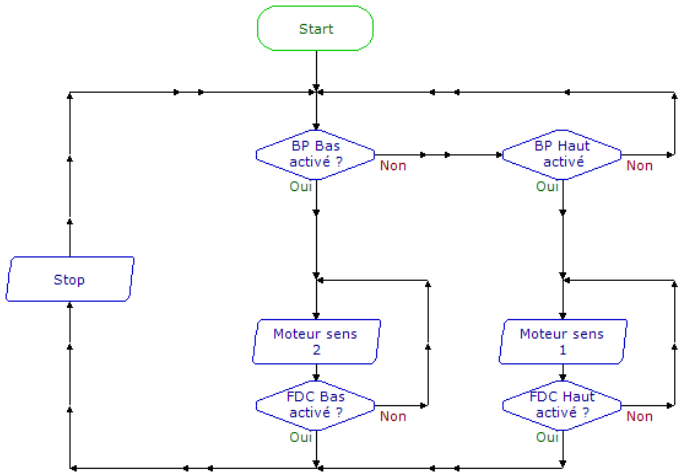
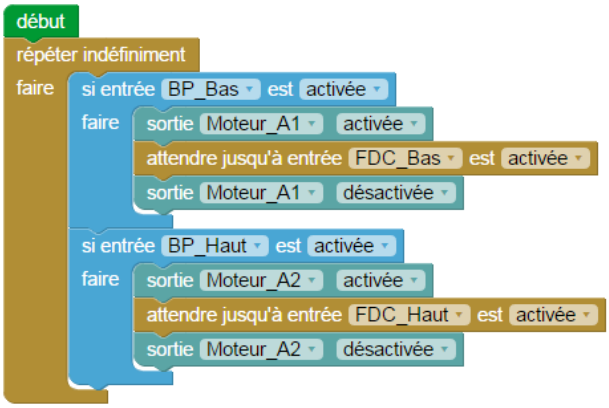
Objectif : contrôler le moteur avec les boutons poussoirs. Monter lors d'un appui sur le bouton du haut, descendre lors d'un appui sur le bouton du bas.

Notion abordée : utilisation des commandes conditionnelles.

Instructions utilisées :



Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_B3_Organigramme.plf	Fichier Blockly : PE_N1_B3.xml

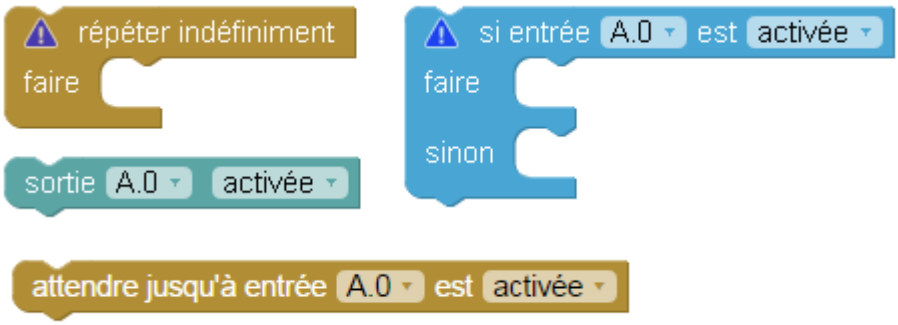
Remarque : Ne pas surcharger le programmes de conditions si, le programme cherchera à vérifier toutes les conditions une à une et une condition pourrait en annuler une autre.
Le programme ne permettra pas deux montées successives.

Exercice niveau 1 - B.4 : Contrôle moteur

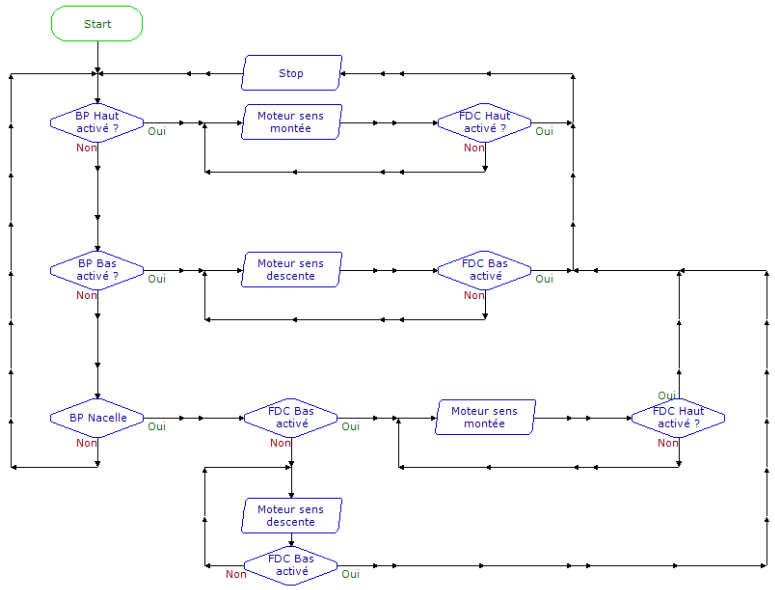
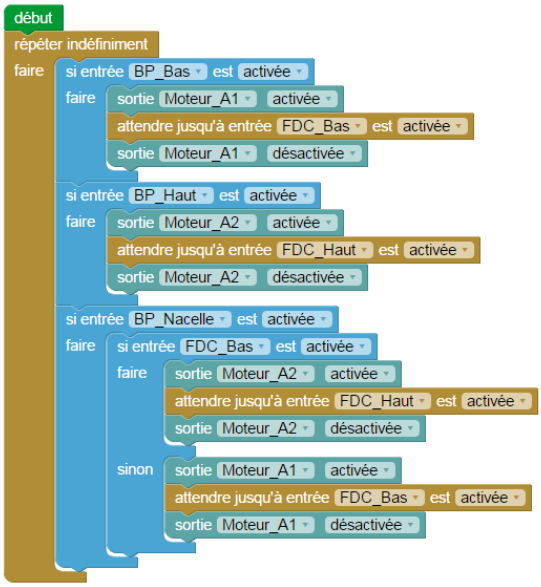
Objectif : contrôler le moteur avec les boutons poussoirs. Monter lors d'un appui sur le bouton du haut, descendre lors d'un appui sur le bouton du bas. Lors d'un appui sur le bouton de la nacelle, monter si la plate-forme était en bas et inversement.

Notion abordée : utilisation des commandes conditionnelles.

Instructions utilisées :



Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_B4_Organigramme.plf	Fichier Blockly : PE_N1_B4.xml

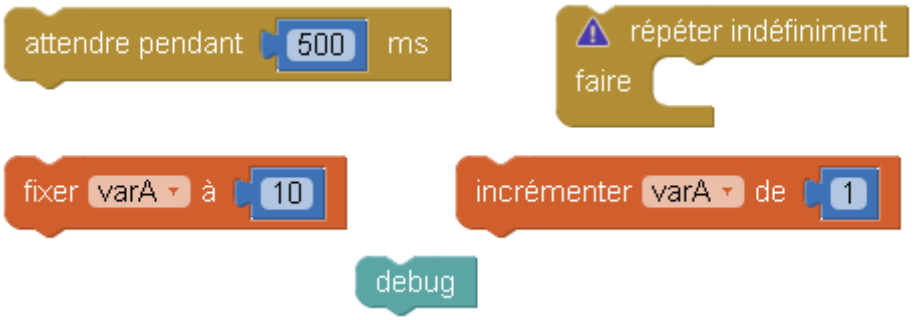
Niveau 1 - C

Exercice niveau 1 - C.1 : Utilisation des variables

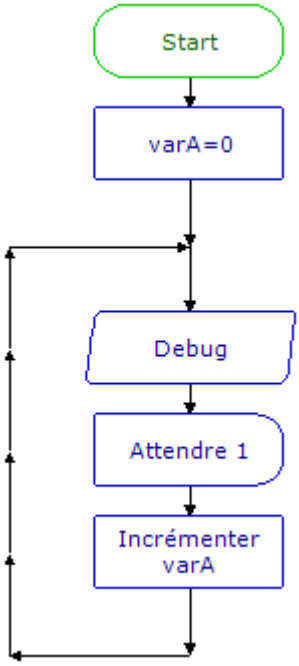
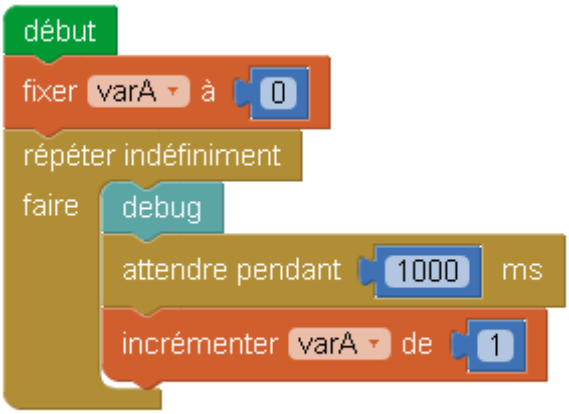
Objectif : incrémenter une variable au cours du temps et observer sa valeur à l'aide du PC (débogage).

Notions abordées : la variable : définition et incrémentation, debug.

Instructions utilisées :



Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_C1_Organigramme.plf	Fichier Blockly : PE_N1_C1.xml

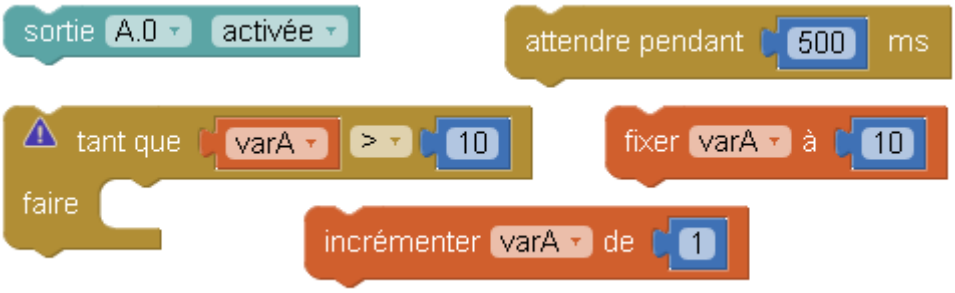
Remarques : la commande « debug » est utilisée afin de retourner la valeur des variables à l'ordinateur. Il est donc indispensable de brancher le câble de programmation à l'ordinateur pour avoir un aperçu de leur valeur.

Exercice niveau 1 - C.2 : Utiliser et tester une variable

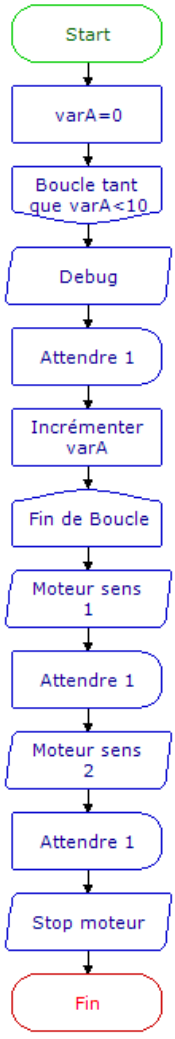
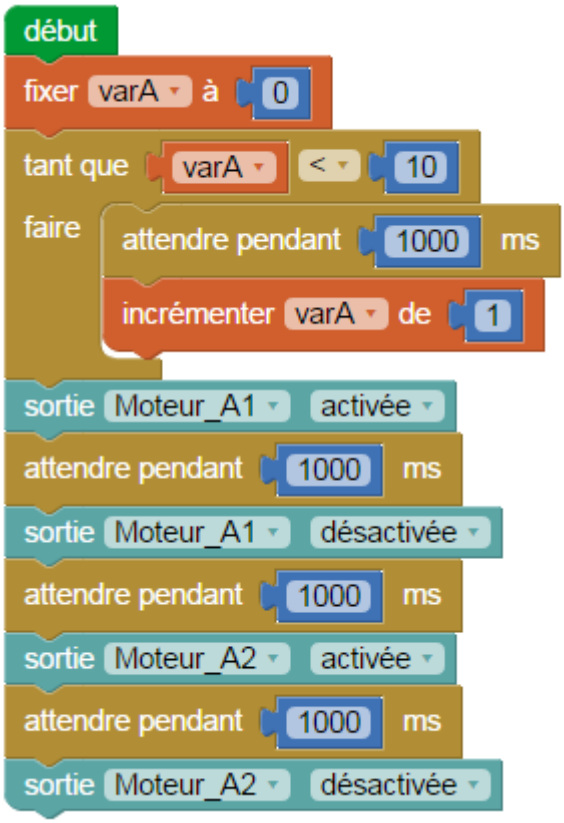
Objectif : incrémenter une variable au cours du temps. Lorsque la variable est supérieure à 10, activer les moteurs et les désactiver

Notion abordée : boucle tant que dépendant d'une variable

Instructions utilisées :



Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_C2_Organigramme.plf	Fichier Blockly : PE_N1_C2.xml

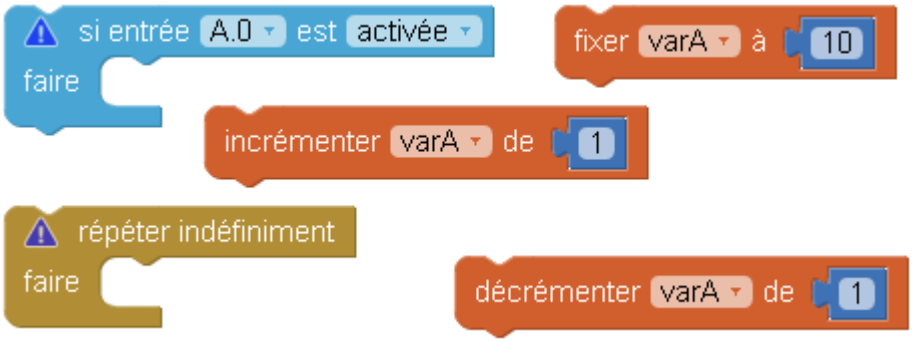
Remarque : cet exercice peut être utilisé comme un minuteur.

Exercice niveau 1 - C.3 : Contrôler la valeur d'une variable à l'aide des boutons poussoirs

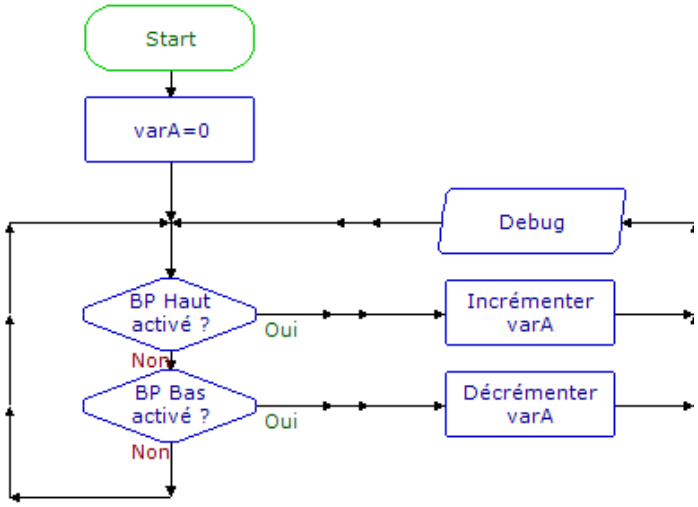
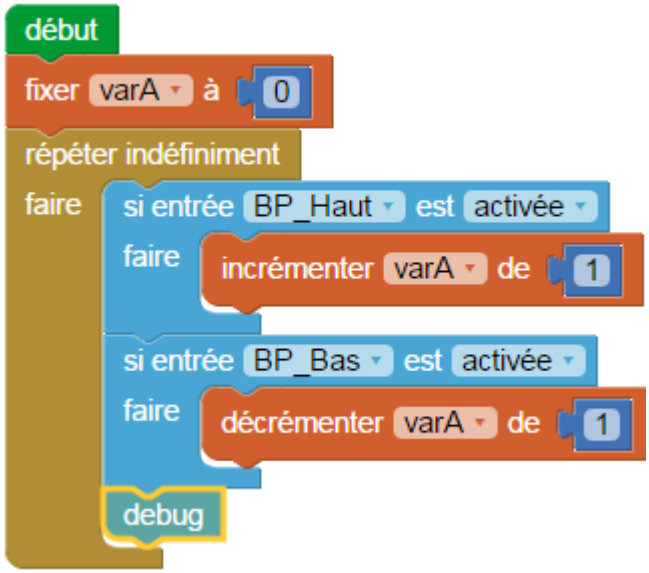
Objectif : incrémenter une variable à l'appui d'un bouton poussoir, décrémenter la même variable à l'appui de l'autre bouton poussoir.

Notions abordées : test sur entrées et incrémentation/décrémentation contrôlée d'une variable

Instruction utilisée :



Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_C3_Organigramme.plf	Fichier Blockly : PE_N1_C3.xml

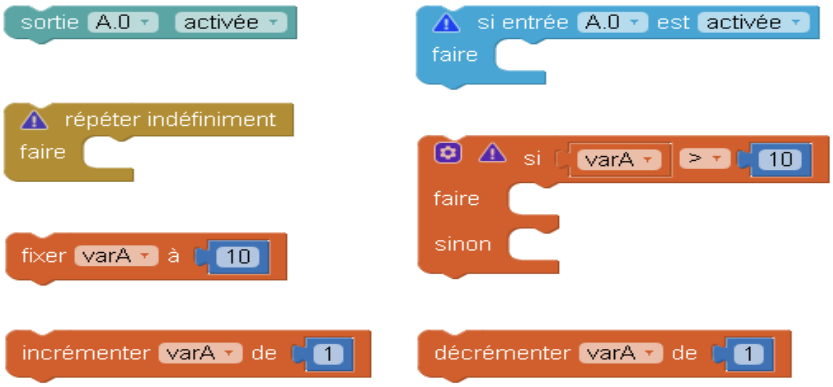
Remarques : deux tests sont insérés l'un après l'autre. La vitesse d'exécution du programme donne l'impression que les commandes sont exécutées en même temps.
La commande « debug » est utilisée afin de retourner la valeur des variables à l'ordinateur. Il est donc indispensable de brancher le câble de programmation à l'ordinateur pour avoir un aperçu de leur valeur.

Exercice niveau 1 - C.4 : Tests /variables

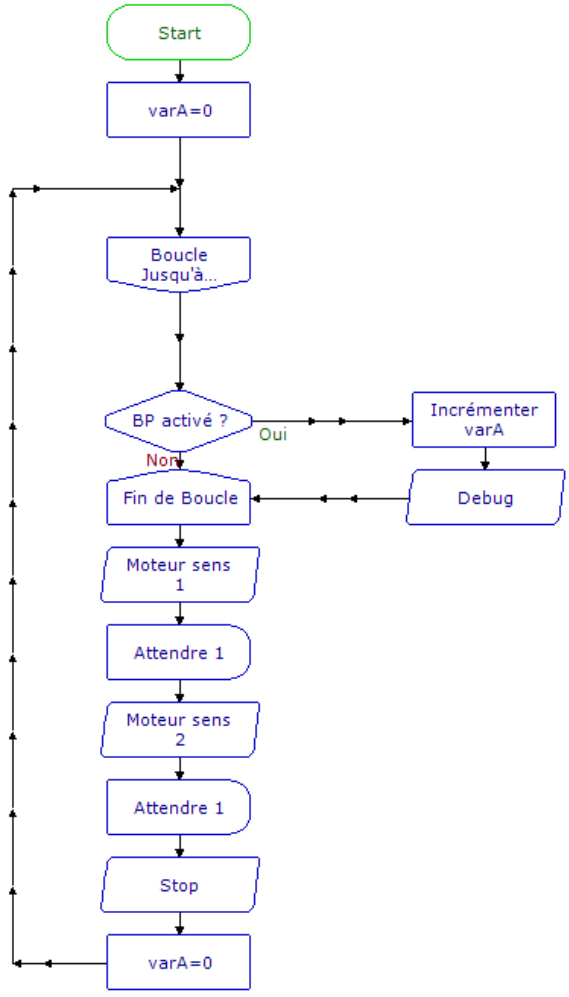
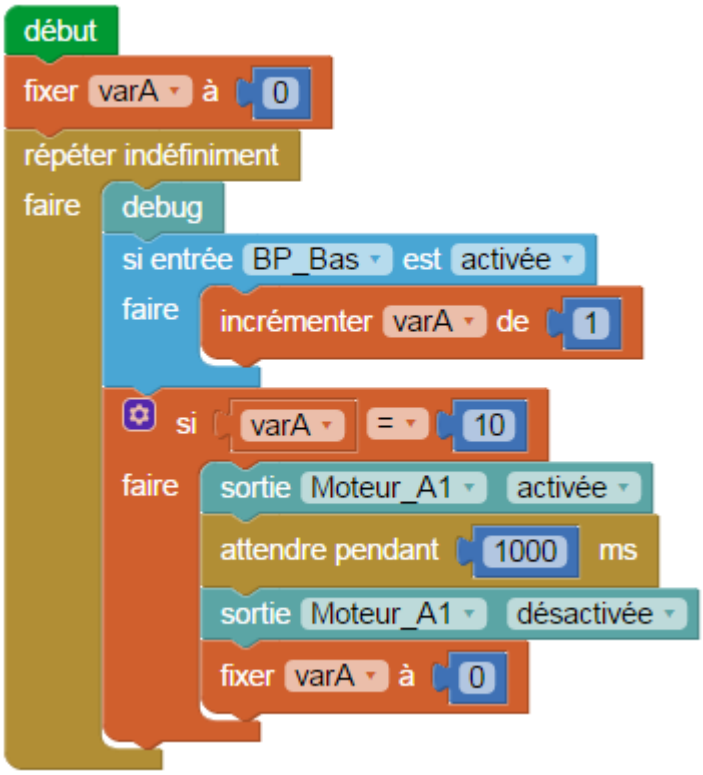
Objectif : incrémenter une variable à chaque appui d'un bouton poussoir. Lorsque le compteur arrive à 10, activer un moteur puis le désactiver après une seconde.

Notion abordée : test dépendant d'une variable

Instructions utilisées :



Correction :

Organigramme	Blocs
	
Fichier organigramme PE6 : PE_N1_C4_Organigramme.plf	Fichier Blockly : PE_N1_C4.xml

Programmation version de base niveau 2

Objectifs :

- 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.

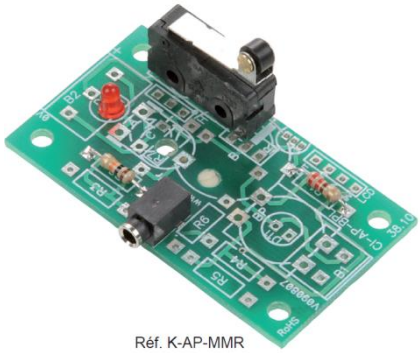
Nom du fichier	**Description**	**Objectif**
Niveau 2 A **Fichier modèle :** PE_BASE.xml		
PE_N2_A1.xml	Utilisation des fins de course	Notions de programmation abordées : -Utilisation des sous-fonctions
PE_N2_A2.xml	Contrôler la montée et la descente avec des boutons-poussoirs.	
PE_N2_A3.xml	Contrôler la montée et la descente avec des boutons-poussoirs et capteur d'arrivée.	

Niveau 2 - A

Exercice niveau 2 - A.1 : montée/descente entre fins de courses

Objectif : Monter et descendre la plateforme avec 2 secondes d'attente entre chaque mouvement. Utiliser les capteurs fins de course pour contrôler l'ouverture et la fermeture.

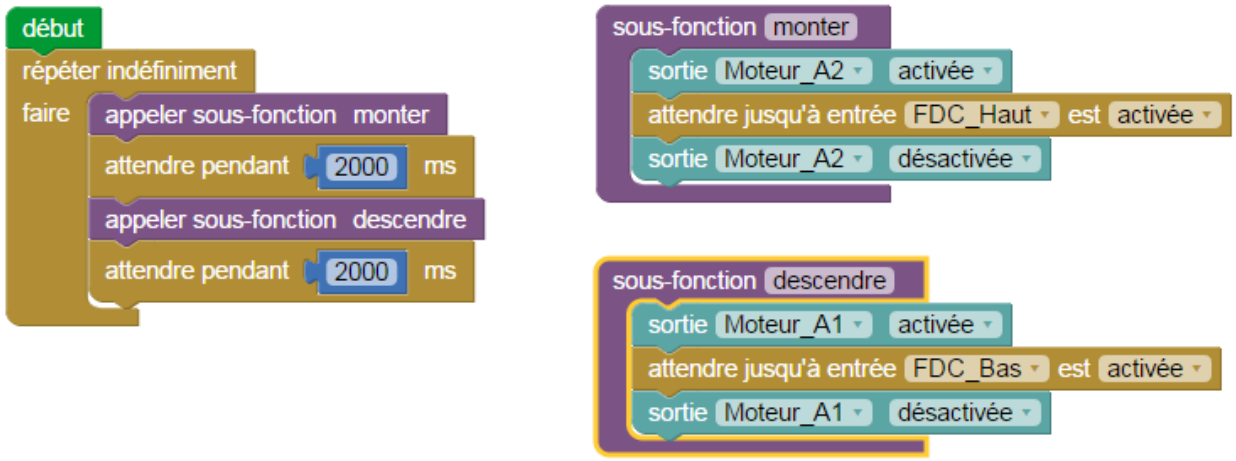
Notions abordées : utilisation des fins de course, procédures (sous-fonctions)



Réf. K-AP-MMR

Correction :

Blocs



Fichier Blockly : PE_N2_A1.xml

Remarque : l'utilisation des sous-fonctions « monter » et « descendre » facilite la lecture du programme.

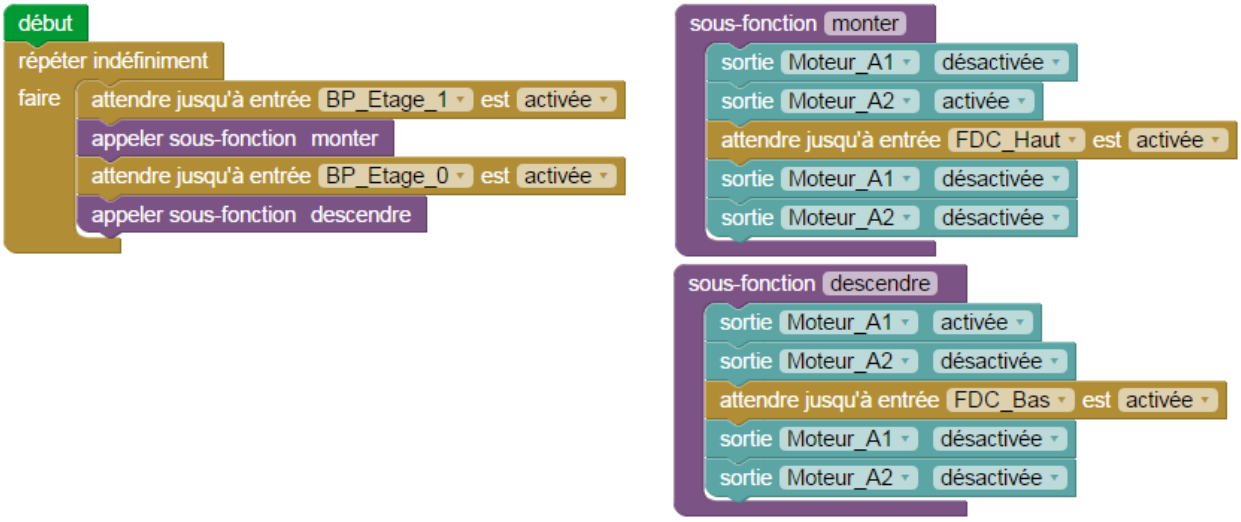
Exercice niveau 2 - A.2 : Contrôle montée/descente

Objectif : Montée de la plateforme à l'appui du bouton haut. Descente de la plateforme à l'appui du bouton bas.

Notions abordées : Réutilisation des sous-fonctions pour un autre programme.

Correction :

Blocs



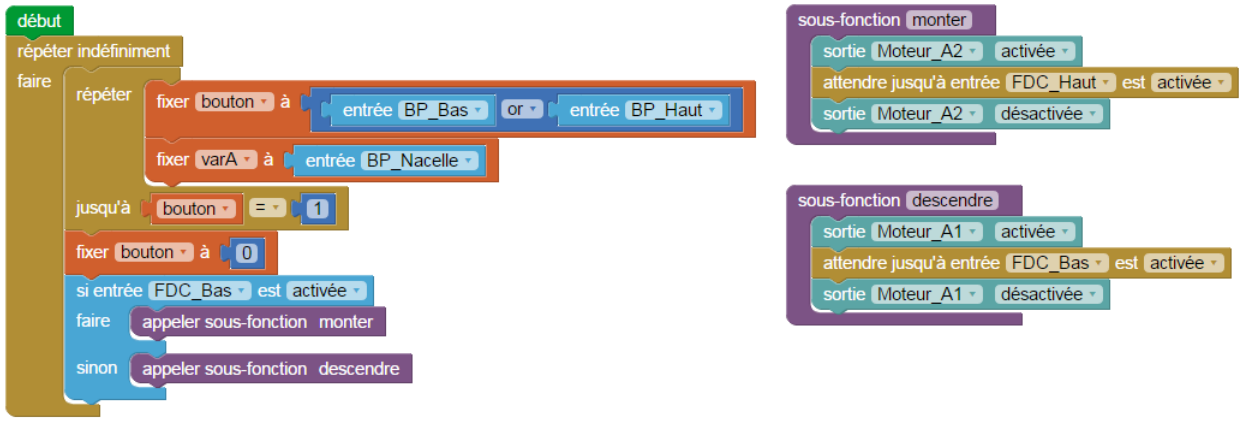
Fichier Blockly : PE_N2_A2.xml

Exercice niveau 2 - A.3 : Contrôle ouverture/fermeture avec BP et capteurs d'arrivée

Objectif : Faire monter et descendre la plateforme à l'aide des BP sans distinction.

Notions abordées : utilisation d'opérateur logique OU (+)

Correction :

Blocs

Fichier Blockly : PE_N2_A3.xml

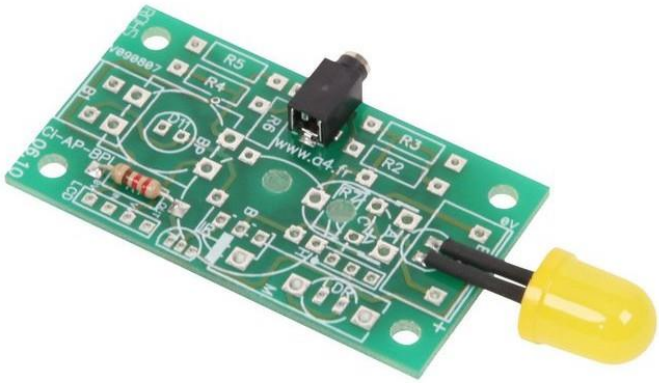
Programmation niveau 3

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 voyant lumineux **Fichier modèle :** PE_BASE.xml		
PE_N3_A1.xml	Allumer le voyant lumineux pendant 3 secondes puis l'éteindre.	Fonctionnalité matérielle abordé : Utilisation de la télécommande IR Notions de programmation abordées : Utilisation d'un block dédié à la communication IR
PE_N3_A2.xml	Allumer le voyant lumineux pendant 3 secondes puis l'éteindre de façon répétée.	
PE_N3_A3.xml	Allumer le voyant lumineux à l'aide d'un bouton poussoir et l'éteindre lorsqu'on relâche ce bouton.	
PE_N3_A4.xml	Reprendre l'exercice PE_N2_A2.xml Ajouter un clignotement de la LED pour chaque mouvement de la plate-forme.	
PE_N3_A5.xml	Reprendre l'exercice précédent. Lors de l'appui du bouton sur la nacelle, la plate-forme descend ou monte en fonction de son étage d'origine.	
Niveau 3 B – Module Bluetooth **Fichier modèle :** PE_BASE.xml		
PE_N3_B1.xml	Contrôler la descente et la montée de la plateforme à l'aide de 2 boutons présent sur l'application Android.	Fonctionnalité matérielle abordée : module Bluetooth Notions de programmation abordées : liaison série (hserin/hserout)
PE_N3_B2.xml	Monter ou descendre la plateforme à partir d'un seul bouton disponible sur l'application Android. La LED doit être activée lors d'un déplacement.	
PE_N3_B3.xml	Jouer une sonnerie sur le Smartphone à partir de l'appui d'un BP du portail.	
PE_N3_B4.xml	Faire monter ou descendre la plateforme à partir de boutons sur une application Bluetooth. Jouer une sonnette lorsque la plateforme s'arrête à un étage.	
PE_N3_B5.xml	Reprendre l'exercice précédent, lorsqu'on appuie sur le bouton poussoir de la balance, envoie une demande de mouvement au smartphone, qui peut décider ou non de mettre en marche la plateforme.	

Option : Module voyant lumineux

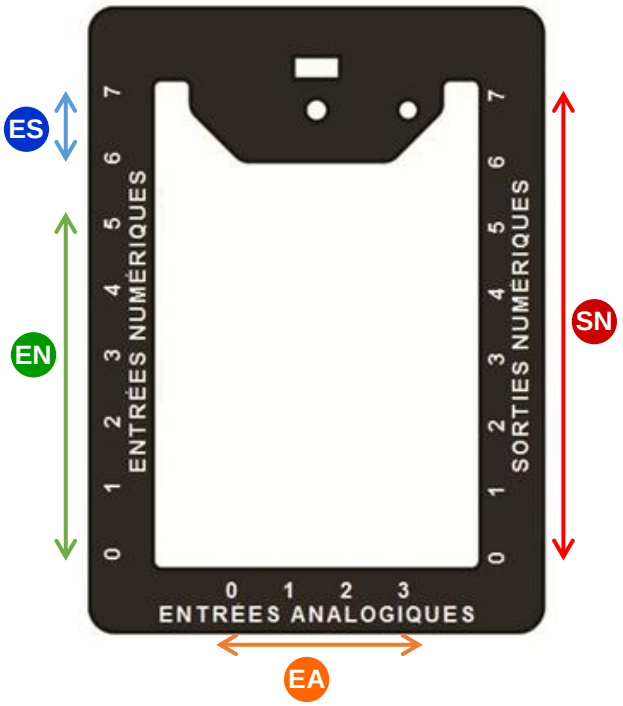


Le module voyant lumineux répond à l'activation d'une sortie.

Si une sortie est activée, le voyant lumineux s'allume.

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	*(libre)*	C.5	
4	Bouton poussoir nacelle	C.4	BP_Nacelle
3	Bouton poussoir haut	C.3	BP_Haut
2	Capteur de fin de course de montée de la plate-forme	C.2	FDC_Haut
1	Capteur de fin de course de descente de la plate-forme	C.1	FDC_Bas
0	Bouton poussoir bas	C.0	BP_Bas
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	Connecté à la broche MOTA-2 de la carte contrôle moteur	B.7	Moteur_A2
6	Connecté à la broche MOTA-1 de la carte contrôle moteur	B.6	Moteur_A1
5	*(libre)*	B.5	
4	*(libre)*	B.4	
3	*(libre)*	B.3	
2	*(libre)*	B.2	
1	*(libre)*	B.1	
0	**Module voyant lumineux**	**B.0**	**Voyant_Lumineux***



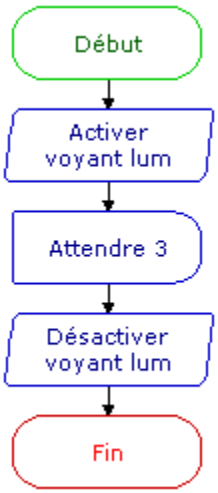
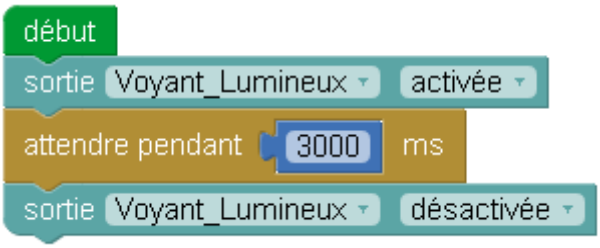
Exercice niveau 3 - A.1 : Activer / désactiver un témoin lumineux

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 Blockly : PE_N3_A1.xml

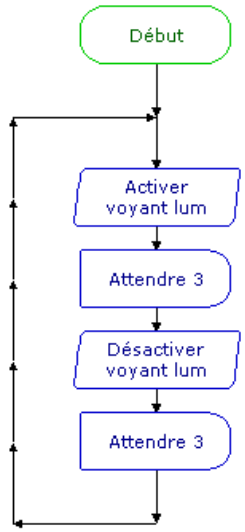
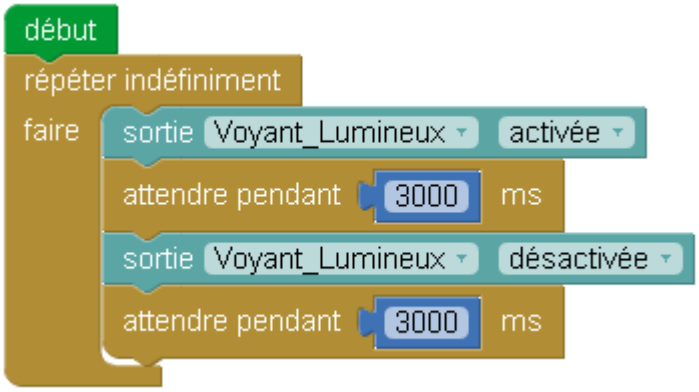
Exercice niveau 3 - A.2 : Répéter une séquence indéfiniment

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

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

Instructions utilisées :

Correction :

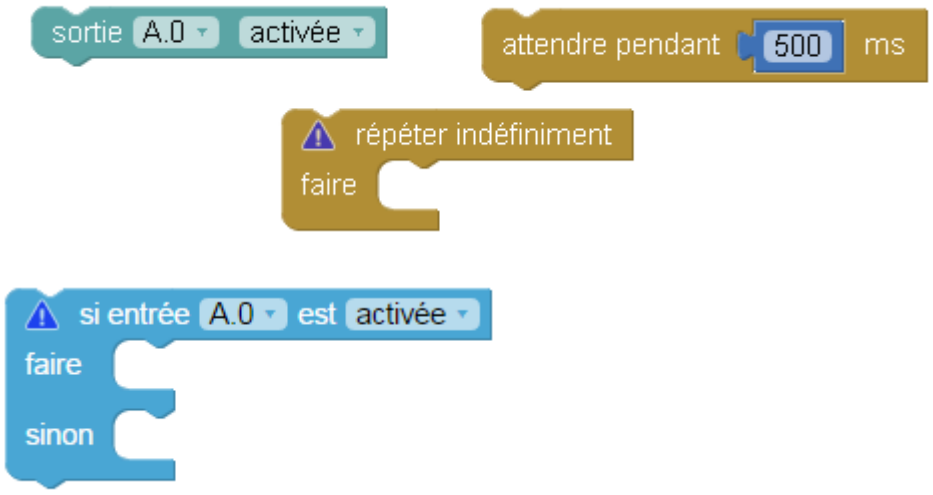
Organigramme	Blocs
	
	Fichier Blockly : PE_N3_A2.xml

Exercice niveau 3 - A.3 : Allumer une LED à l'appui d'un bouton poussoir

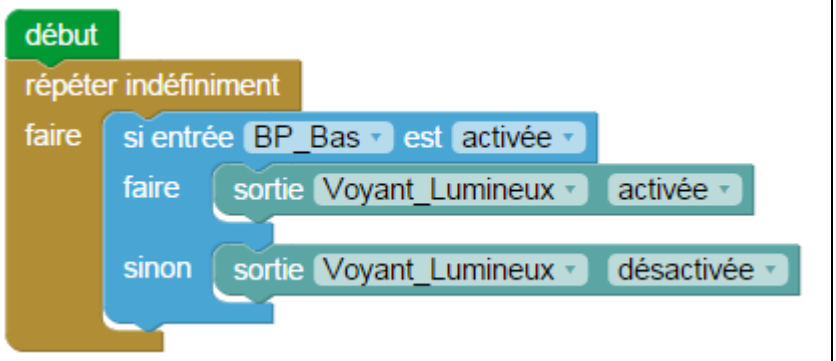
Objectif : allumer le voyant lumineux à l'aide d'un bouton poussoir et l'éteindre lorsqu'on relâche ce bouton.

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

Instructions utilisées :



Correction :

Blocs

Fichier Blockly : PE_N3_A3.xml

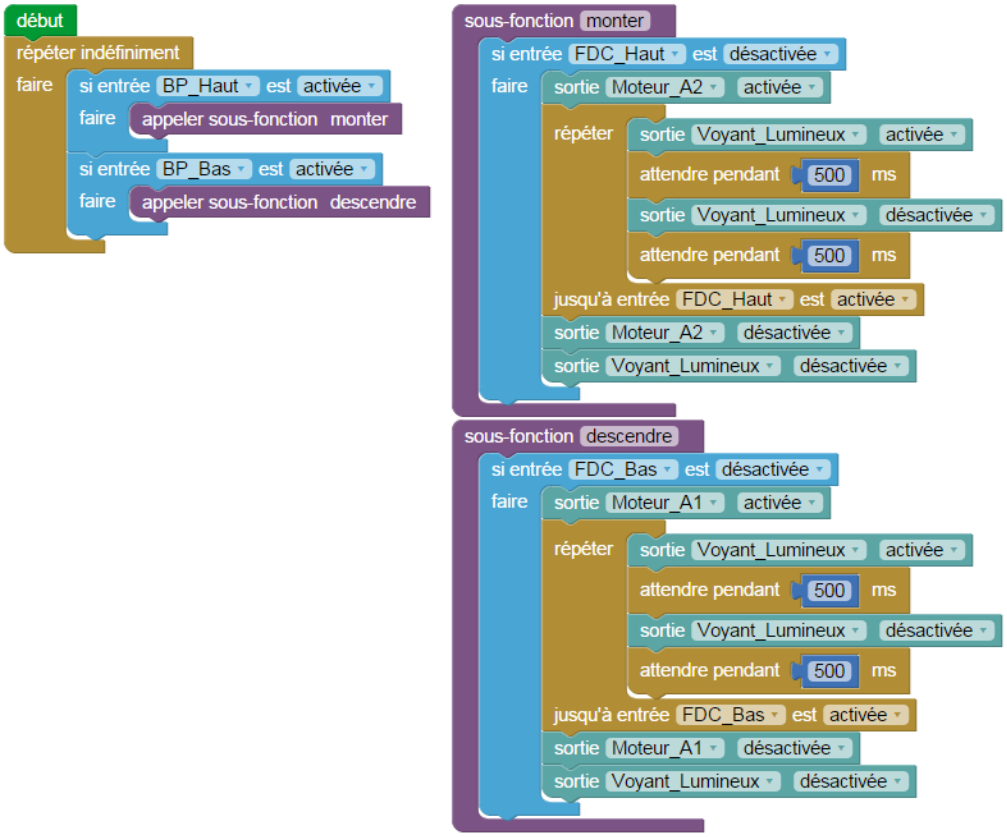
Exercice niveau 3 - A.4 : Allumer une LED lors d'un déplacement de la plateforme

Objectif : Reprendre l'exercice PE_N2_A2.xml Ajouter un clignotement de la LED pour chaque mouvement de la plateforme

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

Correction :

Blocs



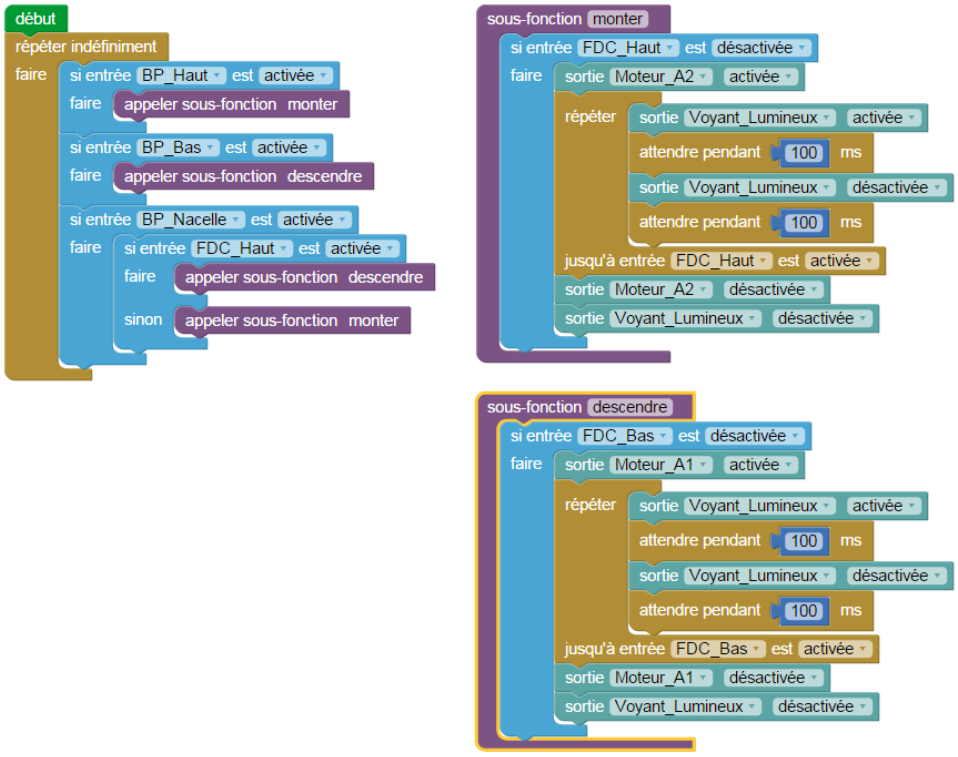
Fichier Blockly : PE_N3_A4.xml

Exercice niveau 3 - A.5 : Finalisation de la plateforme élévatrice

Objectif : Reprendre l'exercice précédent. Lors de l'appui du bouton sur la nacelle, la plateforme descend ou monte en fonction de son étage d'origine.

Correction :

Blocs



Fichier Blockly : PE_N3_A5.xml

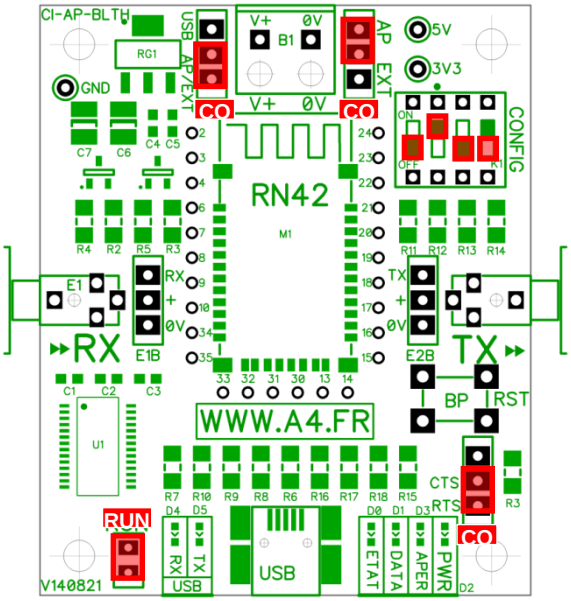
Option : Module Bluetooth

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.

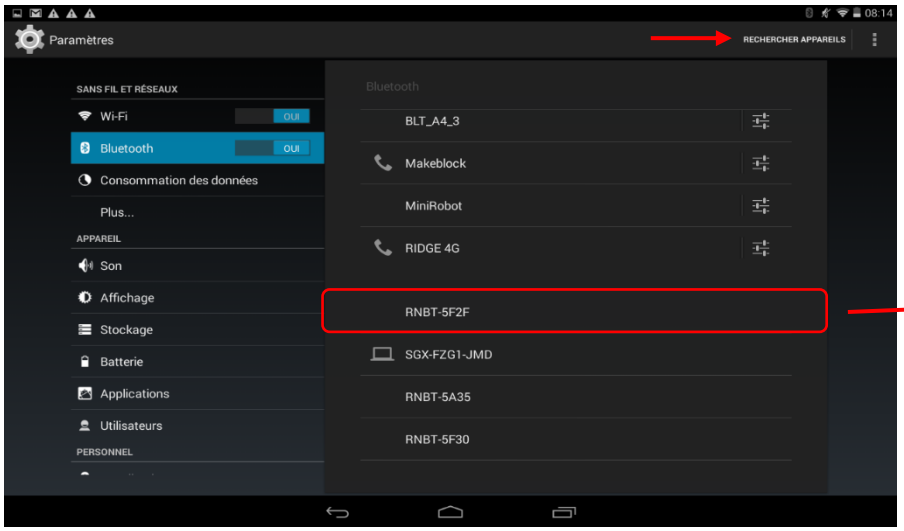
Les interrupteurs **CONFIG** permettent de paramétrer le mode de fonctionnement du module Bluetooth. Ici, l'interrupteur n°2 est positionné sur ON pour sélectionner une vitesse de transmission des données à 9600 bauds.

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.

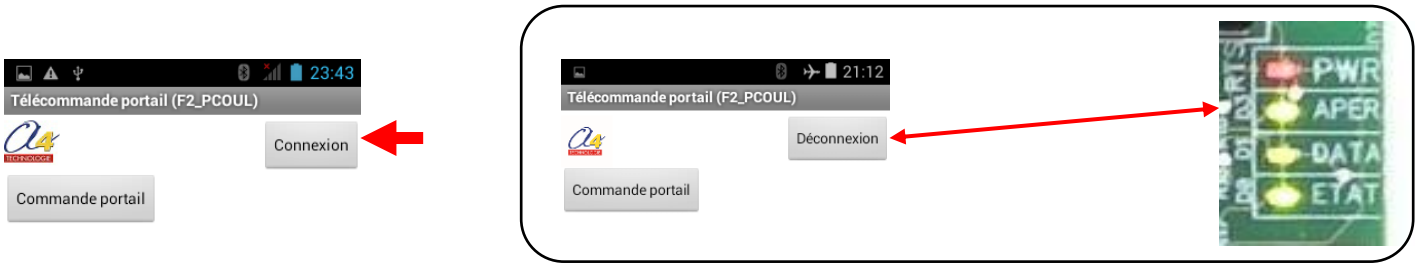
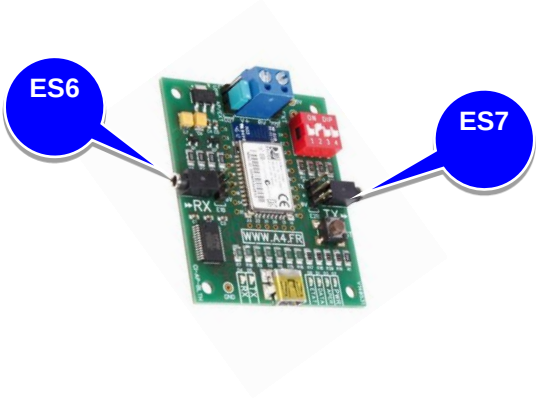


Tableau d'affectation des entrées et sorties en Bluetooth

	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	*(libre)*	C.5	
4	Bouton poussoir nacelle	C.4	BP_Nacelle
3	Bouton poussoir haut	C.3	BP_Haut
2	Capteur de fin de course de montée de la plate-forme	C.2	FDC_Haut
1	Capteur de fin de course de descente de la plate-forme	C.1	FDC_Bas
0	Bouton poussoir bas	C.0	BP_Bas
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	Connecté à la broche MOTA-2 de la carte contrôle moteur	B.7	Moteur_A2
6	Connecté à la broche MOTA-1 de la carte contrôle moteur	B.6	Moteur_A1
5	*(libre)*	B.5	
4	*(libre)*	B.4	
3	*(libre)*	B.3	
2	*(libre)*	B.2	
1	*(libre)*	B.1	
0	Module signal LED rouge rez-de-chaussée	B.0	Voyant_Lumineux

Câblage du module Bluetooth (K-AP-MBLTH)



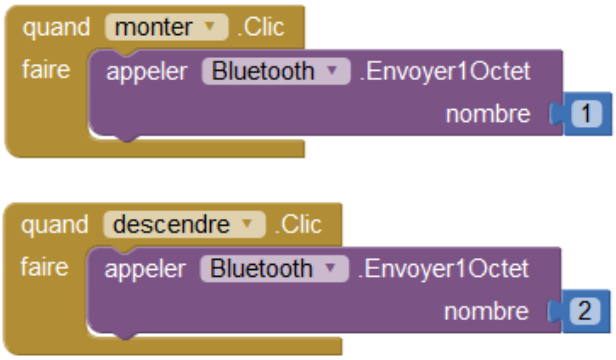
Exercice niveau 3 - B.1 : Monter/descendre avec application Bluetooth

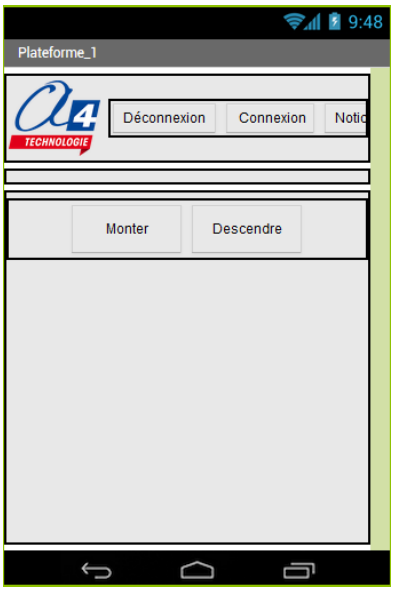
Objectif : Contrôler la descente et la montée de la plateforme à l'aide de 2 boutons présent sur l'application Android.

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

Application Android : Plateforme_1.apk

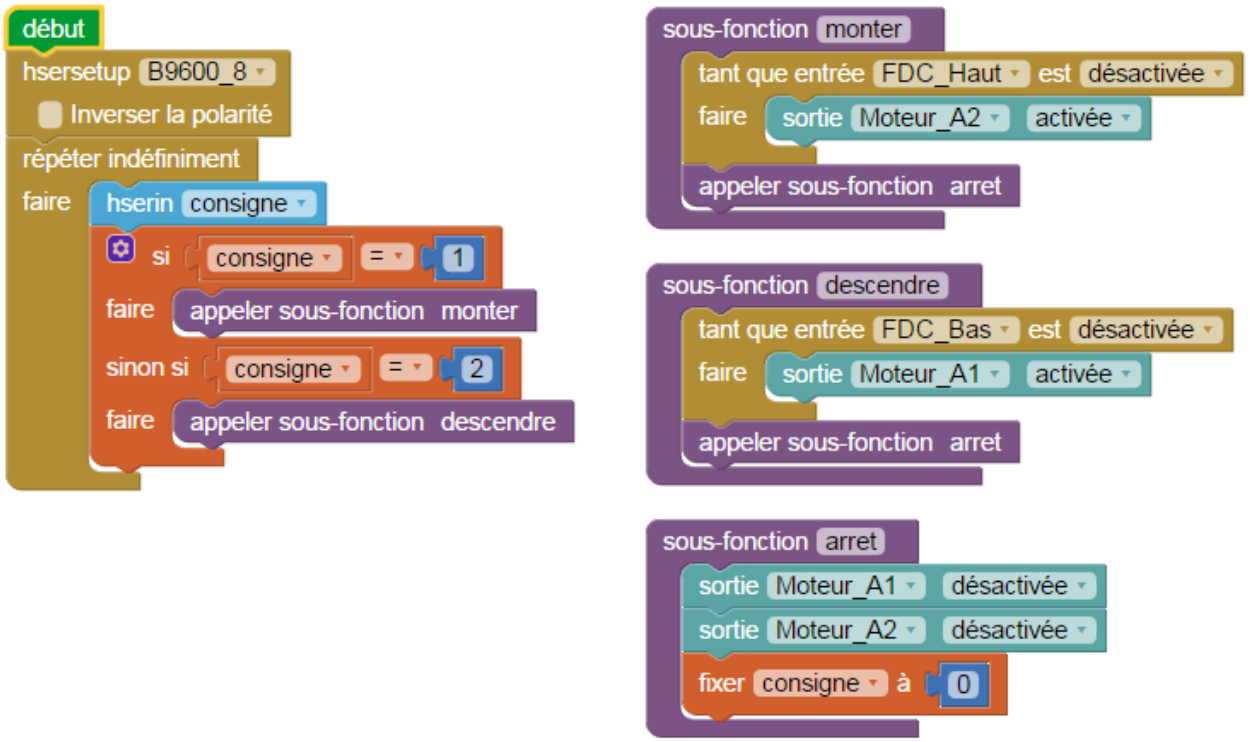
Fichier App Inventor : Plateforme_1.aia





Correction :

Blocs



Fichier Blockly : PE_N3_B1.xml

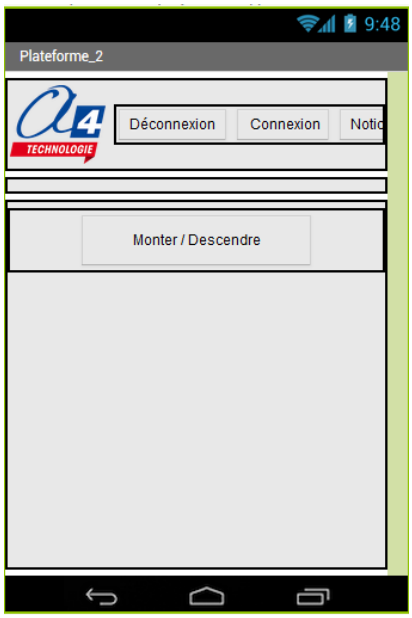
Exercice niveau 3 - B.2 : Contrôle de la plateforme par smartphone

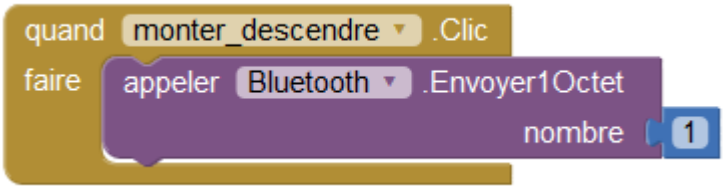
Objectif : Monter ou descendre la plateforme à partir d'un seul bouton disponible sur l'application Android.
La LED doit être activée lors d'un déplacement

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

Application Android : Plateforme_2.apk

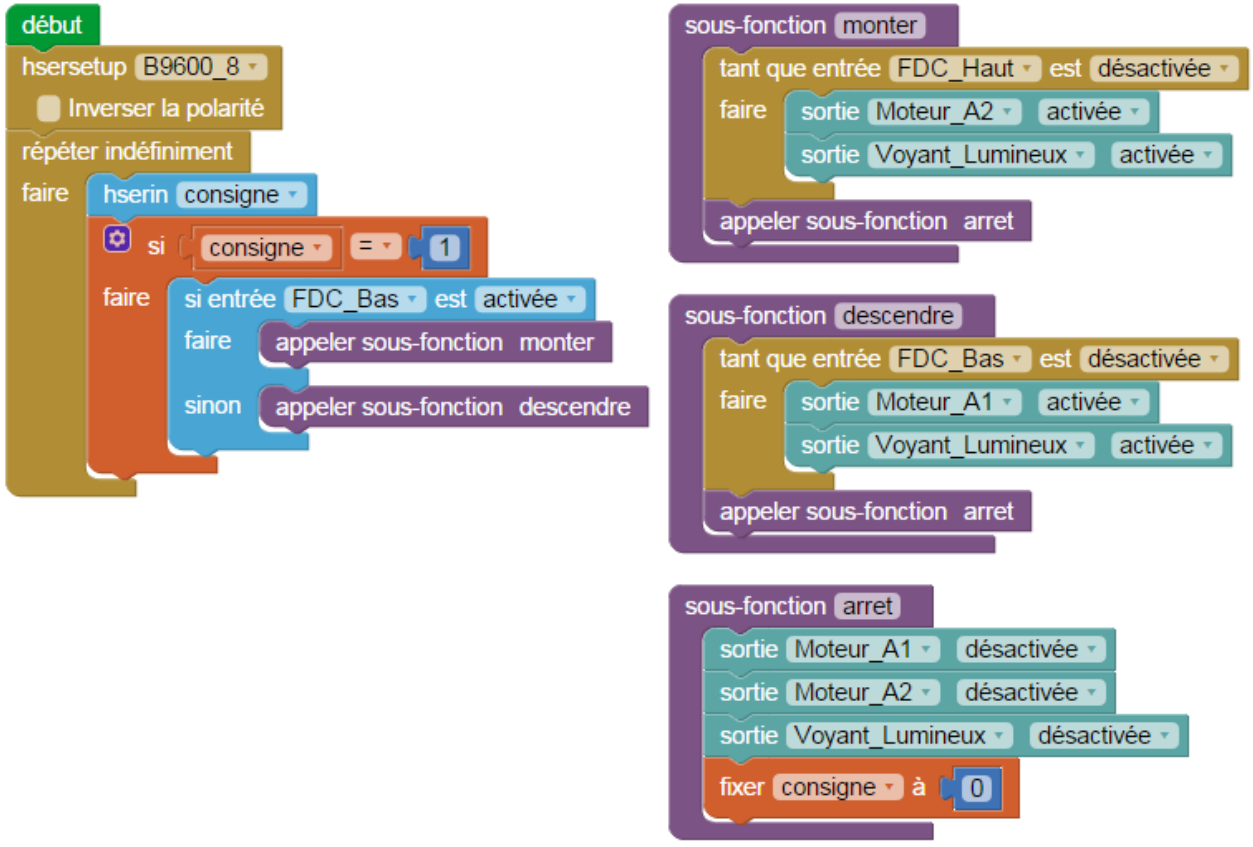
Fichier App Inventor : Plateforme_2.aia





Correction :

Blocs



Fichier Blockly : PE_N3_B2.xml

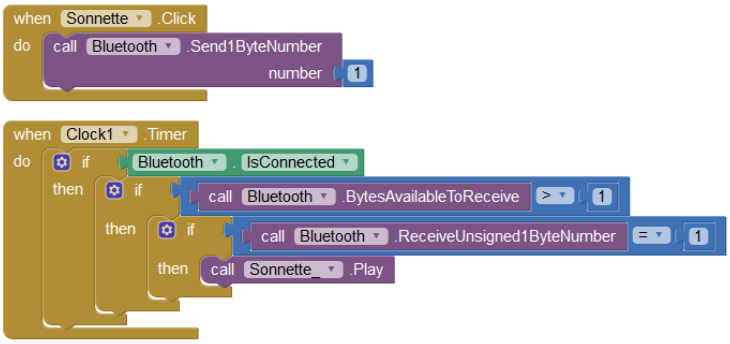
Exercice niveau 3 - B.3 : Envoyer des données vers un Smartphone

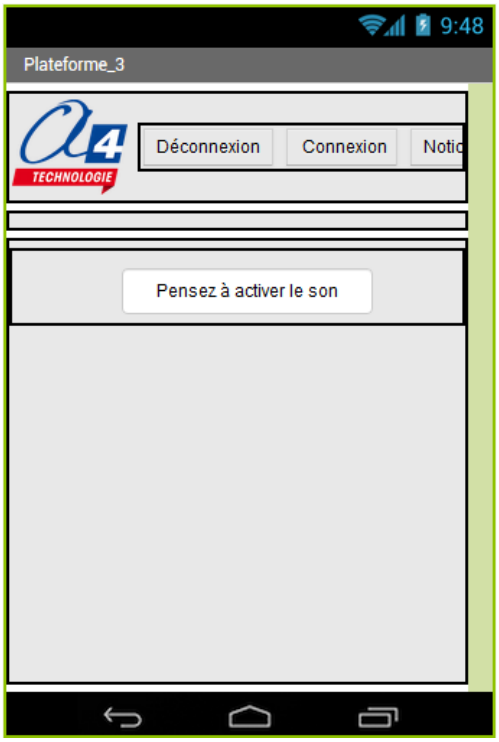
Objectif : jouer une sonnerie sur le Smartphone à partir de l'appui d'un BP du portail ou sur un bouton présent sur l'application

Notion abordée : envoyer des informations à un Smartphone par Bluetooth.

Application Android : Plateforme _3.apk

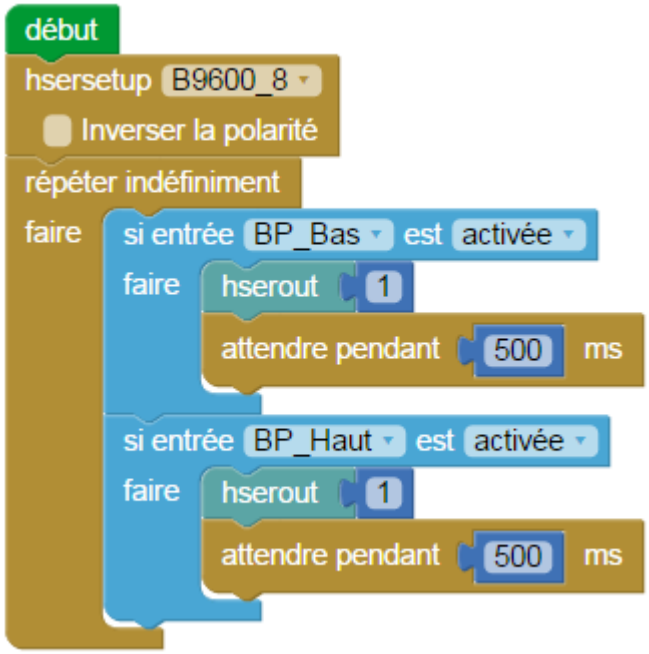
Fichier App Inventor : Plateforme_3.aia





Correction :

Blocs



Fichier Blockly : PE_N3_B3.xml

Remarque : penser à activer le son de votre appareil connecté

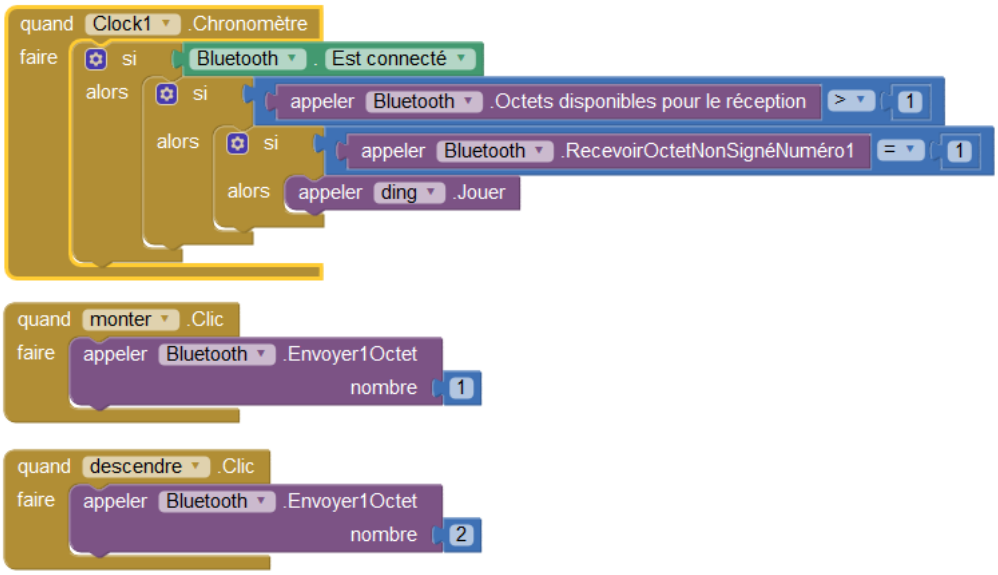
Exercice niveau 3 - B.4 : Envoyer et recevoir des données provenant d'un Smartphone

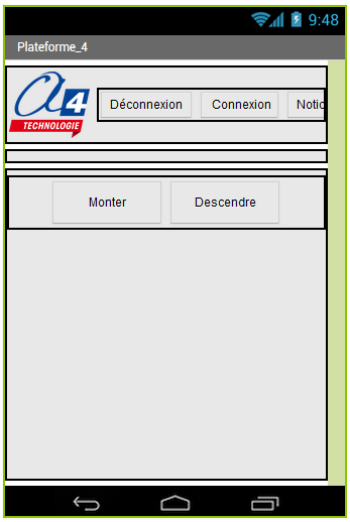
Objectif : Faire monter ou descendre la plateforme à partir de boutons sur une application Bluetooth. Jouer une sonnette lorsque la plateforme s'arrête à un étage.

Notion abordée : envoyer et recevoir des informations à l'aide du module Bluetooth à une application.

Application Android : Plateforme_4.apk

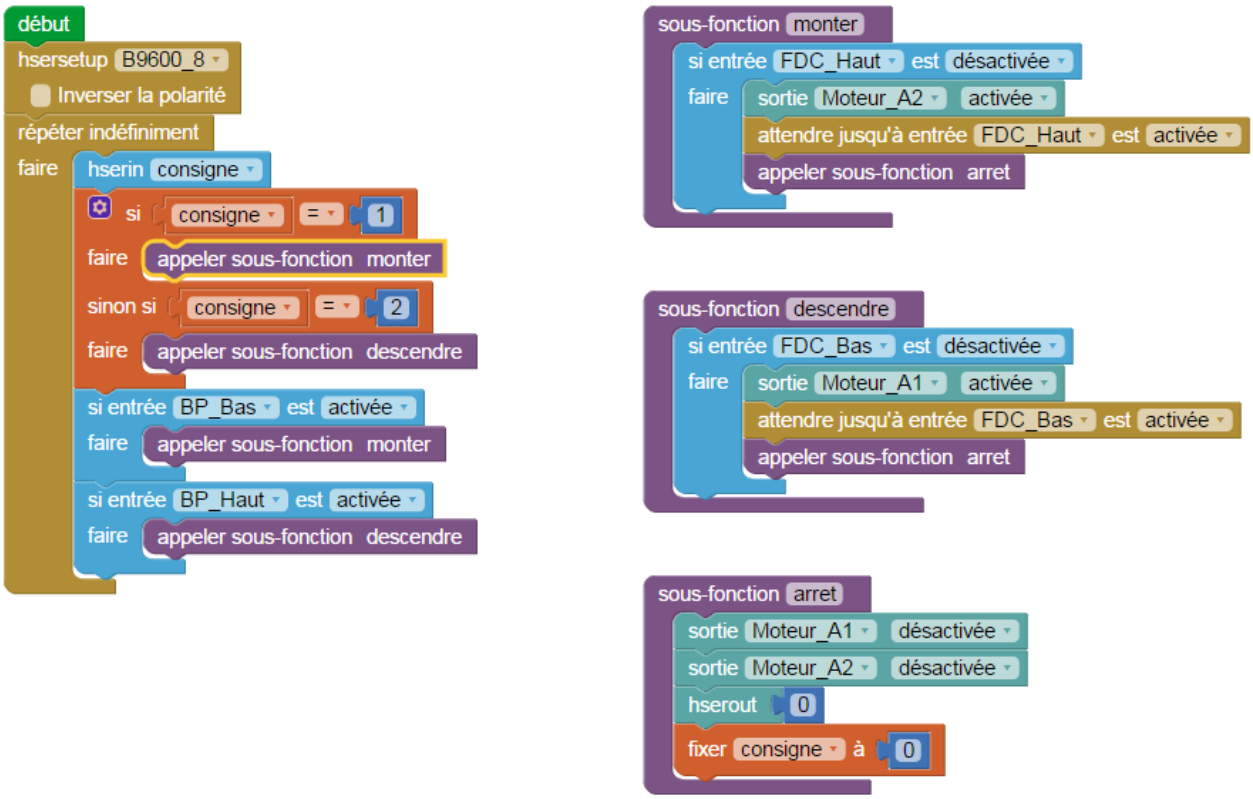
App Inventor : Plateforme_4.aia





Correction :

Blocs



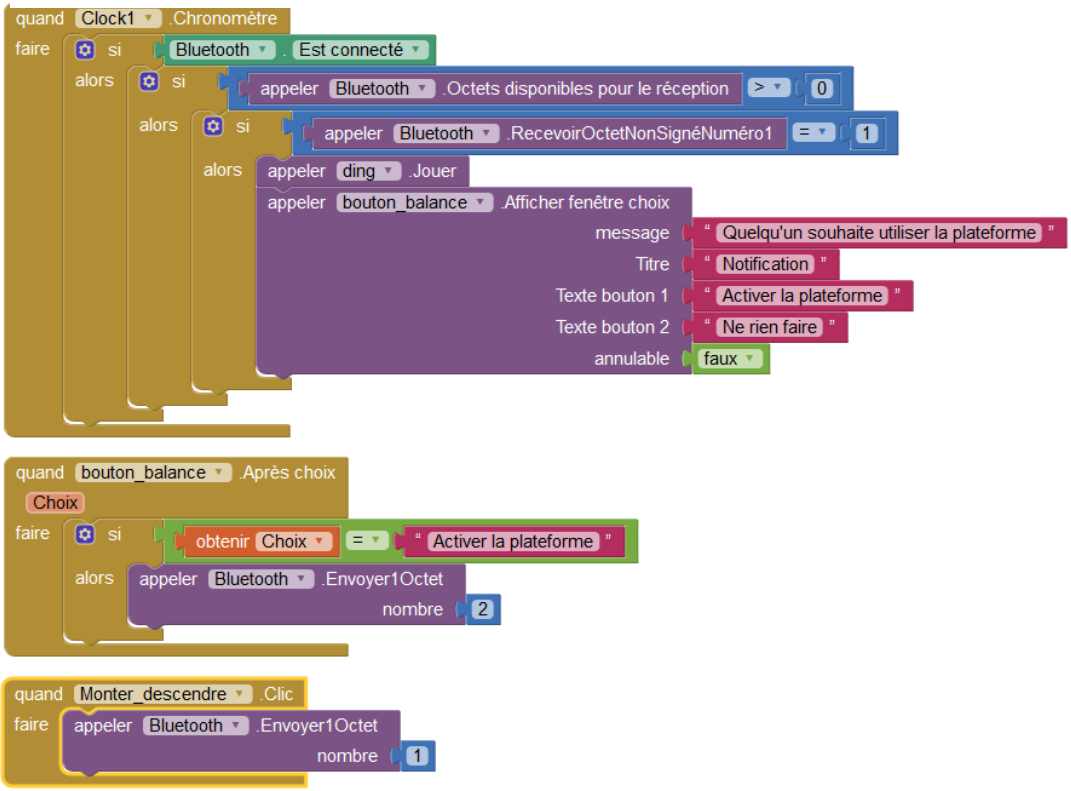
Fichier Blockly : PE_N3_B4.xml

Exercice niveau 3 - B.5 : Gestion à distance d'une information

Objectif : Reprendre l'exercice précédent, lorsqu'on appuie sur le bouton poussoir de la balance, envoie une demande de mouvement au smartphone, qui peut décider ou non de mettre en marche la plateforme.

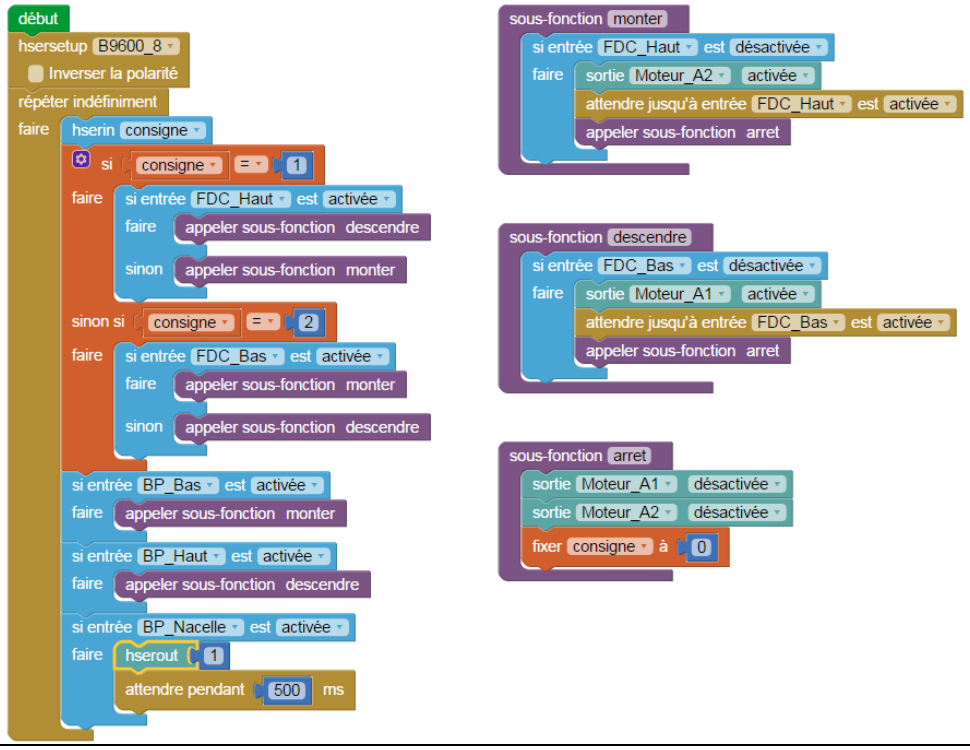
Notion abordée : envoyer et recevoir des informations à l'aide du module Bluetooth à une application.

Application Android : Plateforme_5.apk
App Inventor : Plateforme_5.aia



Correction :

Blocs



Fichier Blockly : PE_N3_B5.xml

Option : Module capteur de courant

Ce module permet de sécuriser le fonctionnement d'un automatisme animé par un moteur à courant continu. En effet, lorsqu'un événement anormal se produit (ex : blocage du portail), la consommation de courant du moteur augmente. La détection de la surintensité au-delà d'un seuil permet de déclencher l'arrêt du moteur et ainsi mettre le système en sécurité.

Le module capteur de courant peut fonctionner selon deux modes.

Mode analogique :
Le cavalier S1 est en position ANA. La sortie MESURE renvoie une tension proportionnelle au courant circulant dans le module. Cette valeur est lue sur une entrée analogique du boîtier.

Mode numérique (tout ou rien) :
Le cavalier S1 est en position NUM. Le potentiomètre SEUIL permet de régler un seuil de courant au-delà duquel la sortie MESURE basculera. Si le courant circulant dans le module dépasse le seuil, la LED témoin et la sortie MESURE s'activent, sinon elles sont inactives.

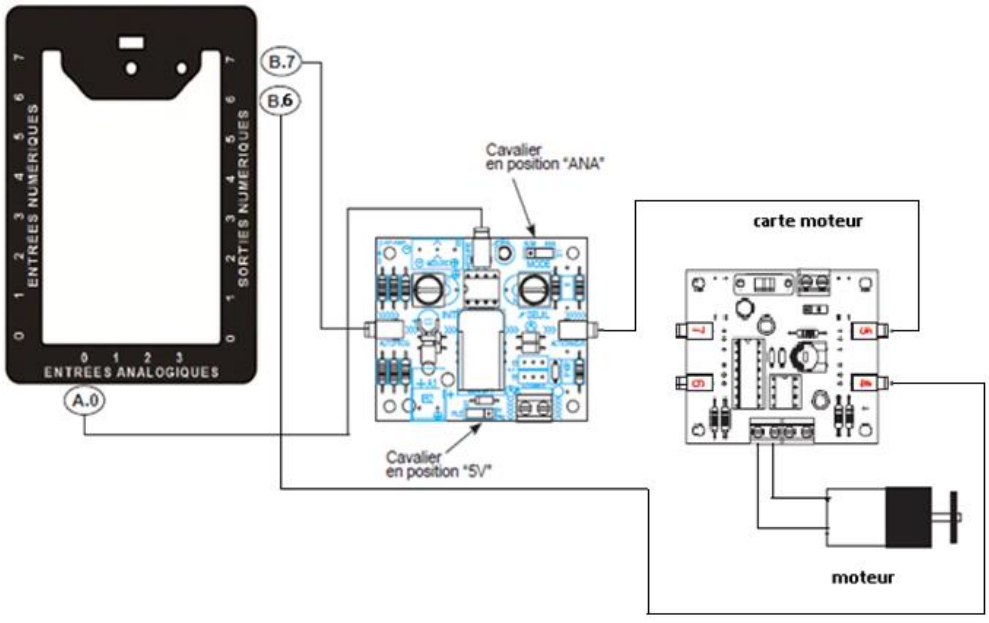
Le potentiomètre INIT permet d'adapter la plage de mesure du module en fonction du contexte d'utilisation (consommation plus ou moins élevé de courant selon matériel utilisé).
La mesure de courant se fait au travers de de la connectique jack (AUTOPROG / ACTIONNEUR) ou bien directement via le bornier à vis ACTIONNEUR.

Mesure via connectique jack :
Le courant mesuré correspond au courant total qui circule dans le montage (courant consommé par l'interface AutoProg, les modules et le moteur).

Mesure via bornier à vis :
Le courant mesuré correspond à celui circulant dans l'élément connecté au bornier (moteur).

Dans la suite des exercices le capteur de courant est utilisé en mode Analogique et la mesure est faite via la connectique jack. Il s'agit de détecter les surintensités dans le moteur ; on considère que la consommation de l'interface et des modules est négligeable par rapport à celle du moteur. Ce mode de d'utilisation facilite le câblage. L'interface AutoProg est alimenté par un bloc d'alimentation externe afin de garantir une tension constante de 5V dans l'ensemble du montage.

Mise en service du module



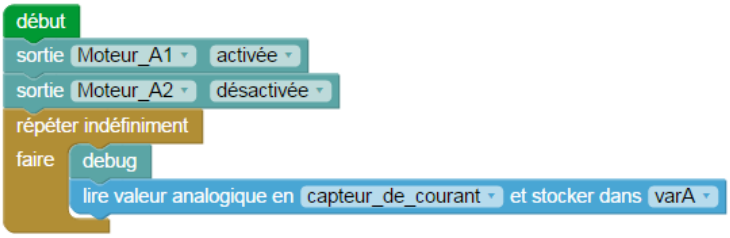
Réglage du potentiomètre d'initialisation :

Il s'agit de régler la tension de sortie du module de telle sorte qu'elle soit maximale lorsque le moteur et bloquée (consommation maximum) afin de travailler sur plage de mesure adaptée au moteur du portail.
En fonctionnement à vide (débrayé du portail) le moteur consomme jusqu'à 250 mA à vitesse moyenne. Si le moteur est bloqué, sa consommation monte au-delà de 500 mA.

Le réglage s'effectue en débrayant le moteur de la barrière (dérailler la barrière pour libérer le moteur de toute contrainte mécanique).

1) Positionner l'interrupteur du module moteur sur OFF puis mettre le cavalier du moteur de Vext sur Vint pour que le moteur soit alimenté directement par l'interface.
   Positionner le cavalier S1 du module Capteur de courant sur ANA et le cavalier S2 sur la position 5V (Schéma)

Relier les embases jack A1 et A2 du module moteur respectivement sur les sorties B.6 et B.7 de l'interface AutoProg.
Relier l'embase jack MESURE du module capteur de courant l'entrée analogique A.0 de l'interface AutoProg puis charger le programme de test **Test_Capteur_Courant.xml**
Ce programme permet de visualiser en direct à l'écran une valeur (varA) qui est proportionnelle à celle du courant consommé par le moteur.



2) Positionner l'interrupteur du module moteur sur ON : le moteur doit tourner
   Ajuster le potentiomètre F-MOTA du module moteur à une position intermédiaire pour fixer la vitesse de rotation du moteur.

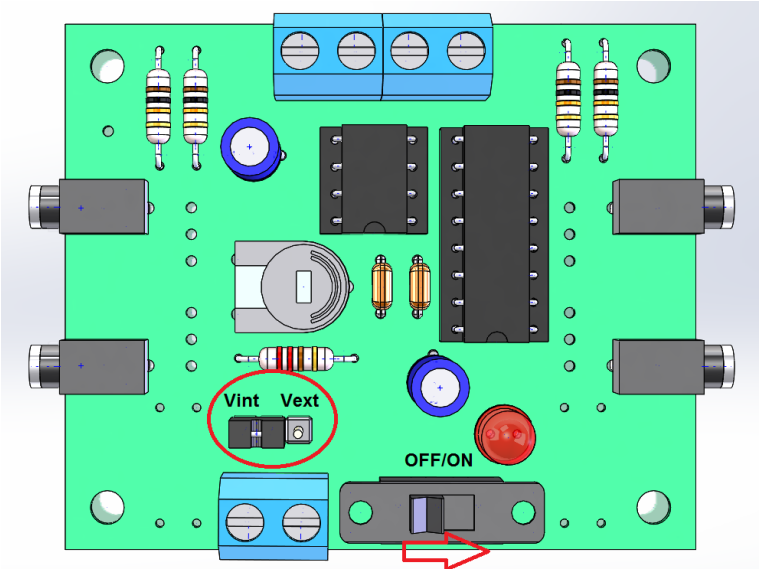
3) Ajuster le potentiomètre MESURE du module capteur de courant jusqu'à avoir une valeur de varA comprise entre 0 et 50.

4)

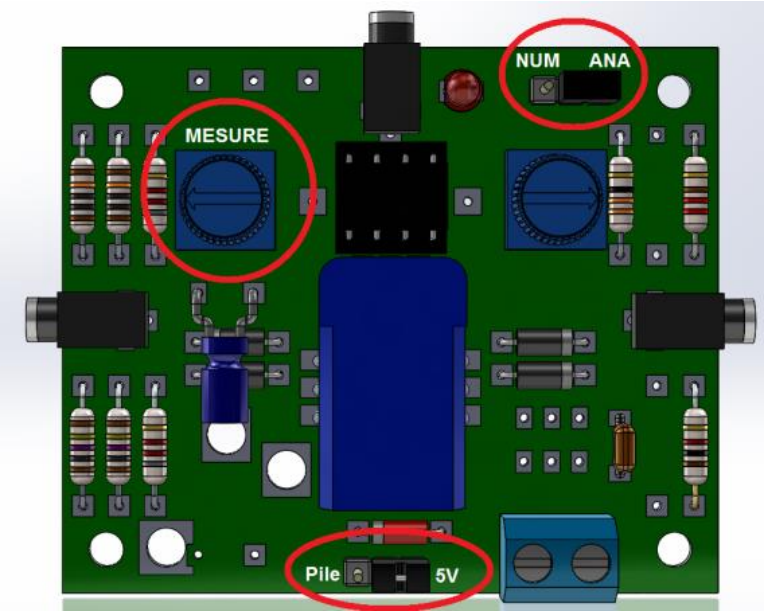
Symbole	Décimal
varA	2

5) Bloquer le moteur manuellement en retenant la roue dentée d'entrainement du portail. La valeur de varA doit augmenter de façon significative (augmentation brutale du courant consommé par le moteur).

   Cette procédure permet de vérifier la bonne configuration du montage et de visualiser la valeur du courant circulant dans le moteur en vue de déterminer le seuil au-delà duquel on considère que celui-ci à consommation excessive (blocage).



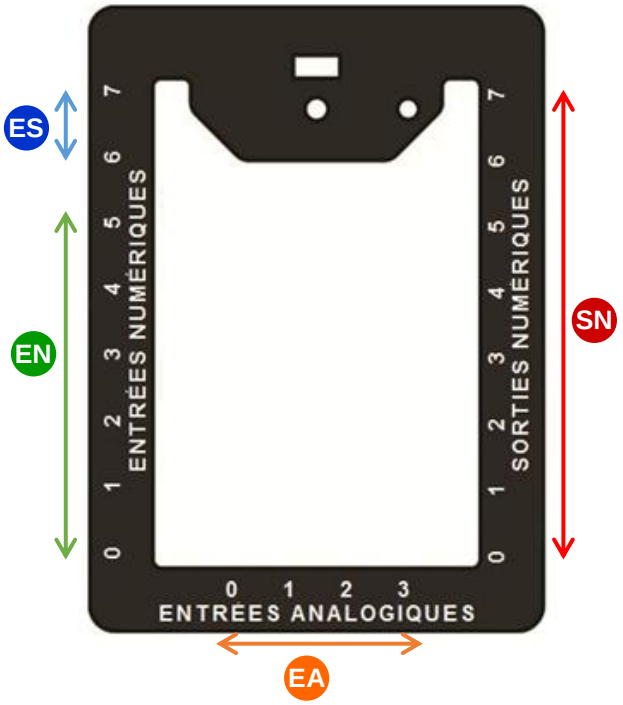
Module pilotage moteurs (REF : K-AP-MMOT-M)



Module Capteur de courant (REF : K-AP-MAMP-M)

Tableau d'affectation des entrées et sorties

	MODULE 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	*(libre)*	C.5	
4	Bouton poussoir nacelle	C.4	BP_Nacelle
3	Bouton poussoir haut	C.3	BP_Haut
2	Capteur de fin de course de montée de la plate-forme	C.2	FDC_Haut
1	Capteur de fin de course de descente de la plate-forme	C.1	FDC_Bas
0	Bouton poussoir bas	C.0	BP_Bas
EA	**MODULES CAPTEURS POUR ENTRÉES ANALOGIQUES**		
3	*(libre)*	A.3	
2	*(libre)*	A.2	
1	*(libre)*	A.1	
0	**Module capteur de courant**	**A.0**	**Capteur_Courant**
SN	**MODULES ACTIONNEURS SORTIES NUMÉRIQUES**		
7	Connecté à la broche MOTA-2 de la carte contrôle moteur	B.7	Moteur_A2
6	Connecté à la broche MOTA-1 de la carte contrôle moteur	B.6	Moteur_A1
5	*(libre)*	B.5	
4	*(libre)*	B.4	
3	*(libre)*	B.3	
2	*(libre)*	B.2	
1	*(libre)*	B.1	
0	Module voyant lumineux	B.0	Voyant_Lumineux*



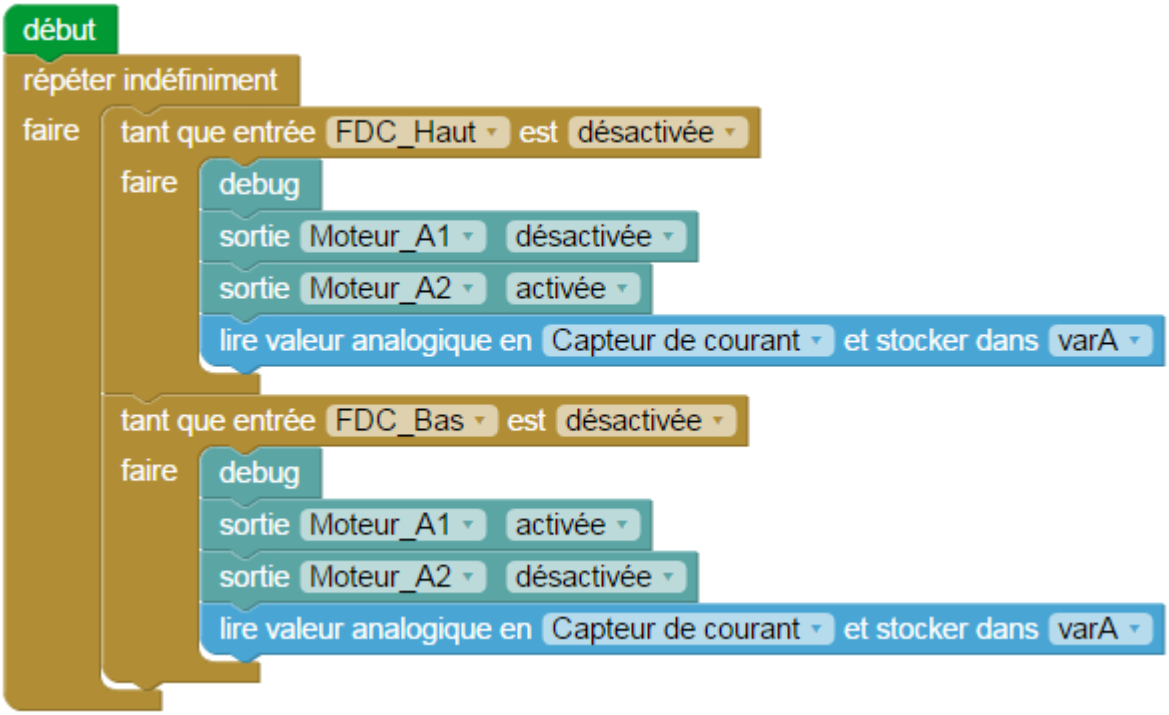
Exercice niveau 3 – C.1 : Utilisation du capteur de courant

Objectif : le portail avance et change de sens à chaque passage sur un capteur fin de course, lire la valeur du capteur de courant en continu et déterminer le seuil (Valeur maximale de varA lorsqu'on bloque la barrière).

Notion(s) abordée(s) : lire une entrée analogique.

Correction :

Blocs

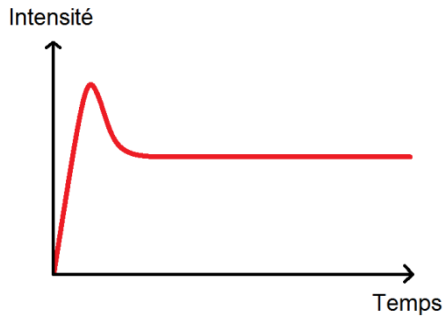


Fichier Blockly : PE_N3_C1.xml

Remarque : Lorsqu'un effort est apporté au moteur, sa demande de courant augmente. Plus la résistance opposante au moteur sera grande, plus il va consommer de courant.

Lorsque d'un démarrage du moteur, il utilise pendant un bref instant plus de courant qu'en fonctionnement normal. Visualiser ce pic d'intensité sur varA à chaque changement de sens et considérer cette valeur comme le seuil bas, cela car le moteur ne doit pas s'arrêter au démarrage à cause de cette surintensité.

Voici une représentation de la surintensité de démarrage du moteur :



Celui-ci après le démarrage va se remettre à la valeur d'intensité normale de fonctionnement.

Il est préférable pour l'exercice suivant de mettre un seuil plus petit que le seuil maximum
Par exemple si lorsque vous bloquez la barrière vous trouvez un seuil max avec varA = 80, nous vous conseillons pour la suite de fixer ce seuil à 70 par exemple.

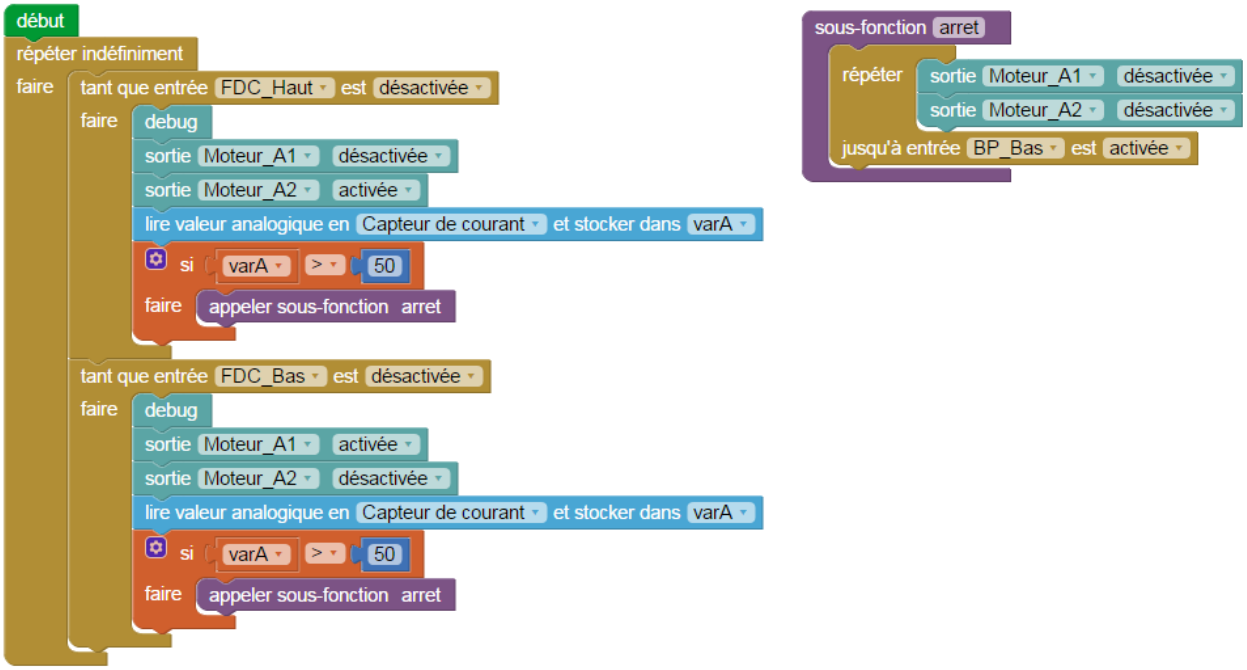
Exercice niveau 3 – C.2 : Capteur de courant et variable

Objectif : à partir de la valeur seuil établie dans l'exercice précédent, arrêter le moteur et lorsque le capteur de courant détecte un blocage (Donc lorsque varA atteint le seuil établit en fin d'exercice N3_C1). Réactiver le programme sur l'appui d'un bouton poussoir.

Notion abordée : lire et interpréter une donnée d'une entrée analogique.

Correction :

Blocs



Fichier Blockly : PE_N3_C2.xml

Remarque : La valeur de seuil sélectionnée ici est 50, mais elle correspondra à celle que vous aurez établit dans votre programme.

Lorsqu'on place une masse sur la plateforme, le courant nécessaire va également augmenter. Il ne faut donc pas mettre une charge trop lourde si vous souhaitez en placer une.

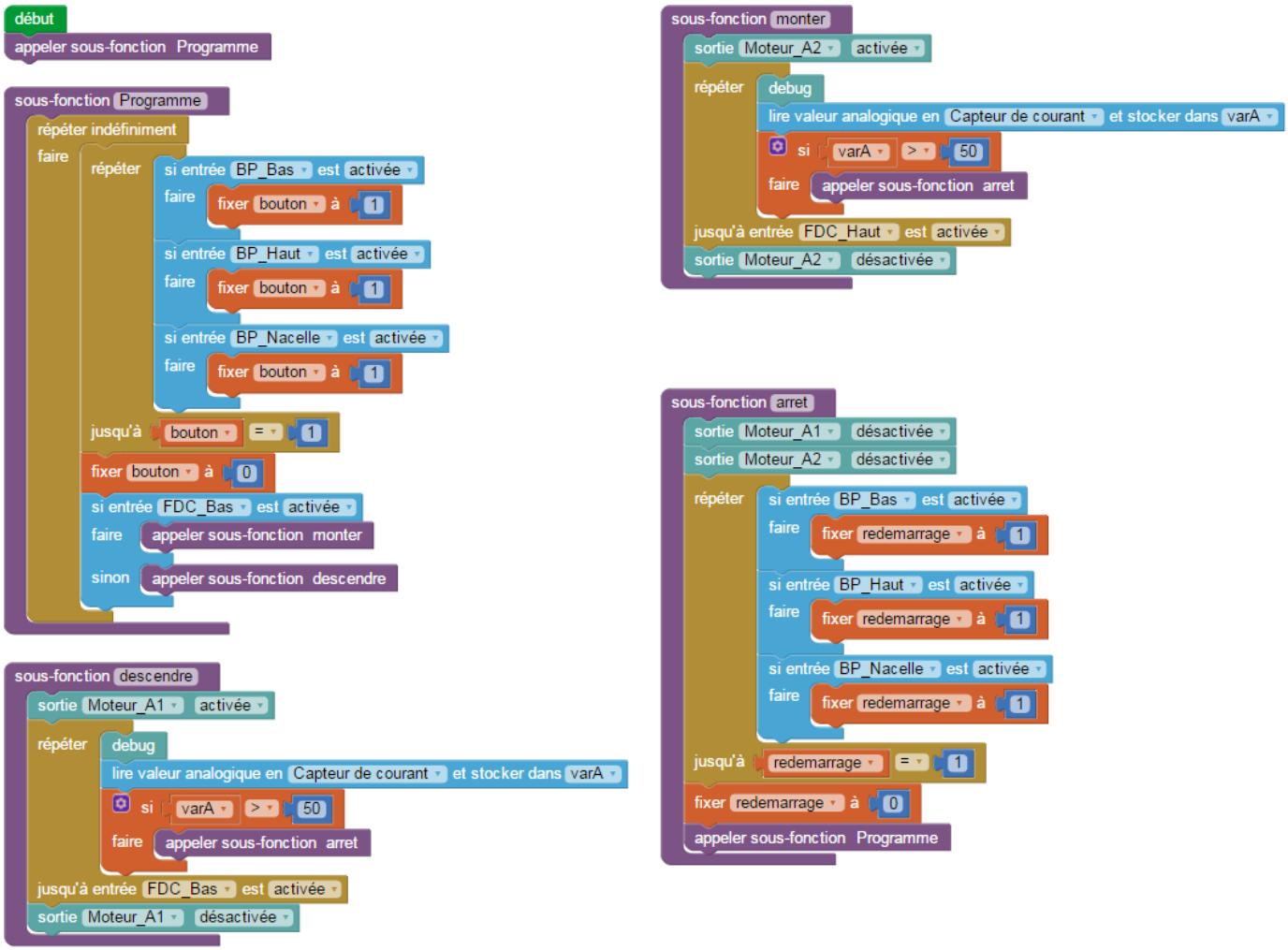
Exercice niveau 3 – C.3 : Capteur de courant et variable

Objectif : Reprendre l'exercice PE_N2_A3.xml
Rajouter la sécurité du capteur de courant et rajouter le BP de la Nacelle pour activer le programme (utiliser des conditions Si)

Notion abordée : lire et interpréter une donnée d'une entrée analogique.

Correction :

Blocs



Fichier Blockly : PE_N3_C3.xml

CONCEPTEUR ET FABRICANT DE MATÉRIELS PÉDAGOGIQUES