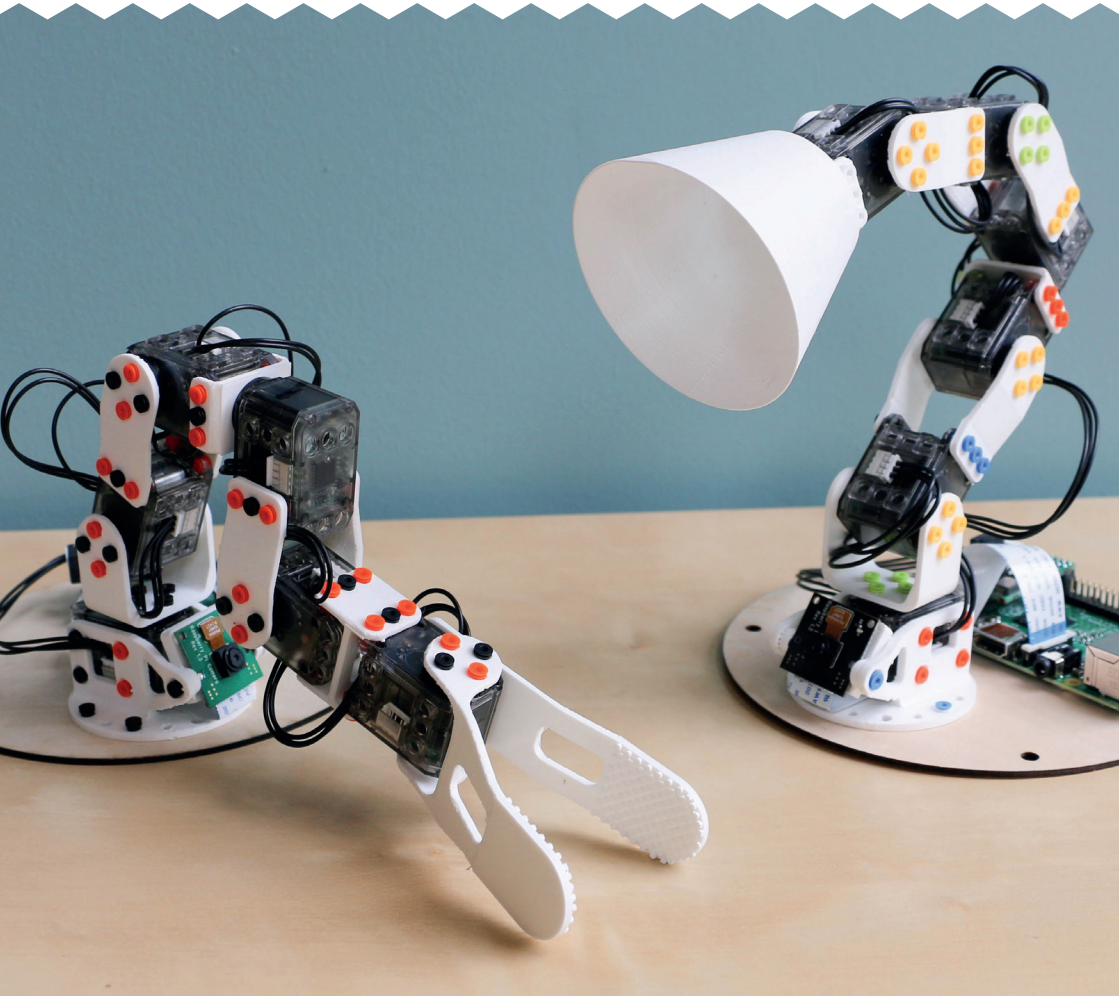
Livret pédagogique : Apprendre à programmer Poppy Ergo Jr en Snap!

Stephanie Noirpoudre, Didier Roy, Marie Demangeat, Thibault Desprez, Théo Segonds, Pierre Rouanet, Damien Caselli, Nicolas Rabault, Matthieu Lapeyre, Pierre-Yves Oudeyer

Apprendre à programmer Poppy Ergo Jr en Snap!

Ce livret propose des activités et des petits défis à réaliser pour se familiariser avec le robot Poppy Ergo Jr et le langage de programmation Snap!.



for Education poppy

N'hésitez pas à nous rejoindre sur notre forum (forum.poppy-project.org, catégorie Education) pour partager vos productions et poser toutes les questions que vous souhaitez !

Apprendre à programmer Ergo Jr en Snap!

Livret d'accompagnement du robot Poppy Ergo Jr

Version 1.0 (rev 0)

Le robot Poppy Ergo Jr a été créé dans le cadre du projet Poppy Éducation de l'Équipe de recherche FLOWERS (Inria, ENSTA Paris Tech), soutenu par la Région Aquitaine et les Fonds Européens FEDER.

Ont contribué au livret :

• Conception et réalisation : Équipe FLOWERS (Inria, ENSTA Paris-Tech) : Stéphanie Noirpoudre, Didier Roy, Marie Demangeat, Thibault Desprez, Théo Segonds, Pierre Rouanet, Damien Caselli, Nicolas Rabault, Matthieu Lapeyre, Pierre-Yves Oudeyer. Sites Web : flowers.inria.fr et www.poppy-project.org

• Design graphique et mise en page : Antonin + Margaux

Remerciements

Au rectorat de l'Académie de Bordeaux et aux enseignants du groupe de travail du projet Poppy Education qui ont aidé à concevoir et à tester les activités et les robots en classe (Said Benrahho, Youcef Bouchemoua, Christophe Casseau, Olivier Éloi, Armelle Grenouilleau, Nicolas Griffe, Gilles Lassus, Georges Layris, Sébastien Prouff, Joël Rivet, Thierry Salem, Sylvain Soulard, Luc Vincent). Génération robots pour leur soutien au projet Poppy.

Sommaire

4. Introduction

6. Commencer en Snap!

10. Activités pour apprendre à programmer Ergo Jr en Snap!

- ***11.*** Contrôler Poppy Ergo Jr
- ***19.*** Programmer par démonstration
- ***25.*** Utiliser la répétition
- ***31.*** Créer son propre bloc Snap!
- ***37.*** Si...alors.. !
- ***45.*** Le bloc For
- ***49.*** Les variables

54. Idées d'activités

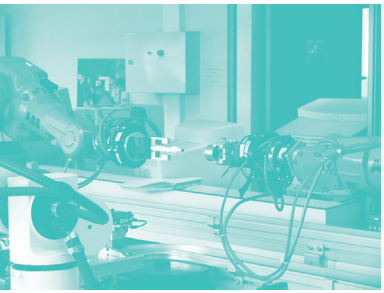
58. Annexe

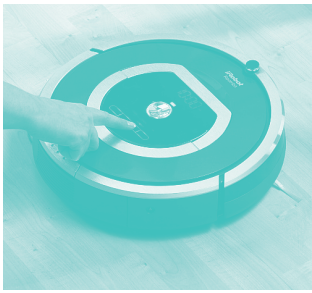
- ***58.*** Liste de mots (activité *Programmer par démonstration*)
- ***59.*** QR codes

Introduction

Qu'est-ce qu'un robot ?

Un robot est une machine dotée de moteurs qui lui permettent de bouger et/ou agir sur son environnement, de capteurs qui lui permettent de percevoir son propre état et son environnement, et d'un système électronique ou informatique qui contrôle ce qu'effectue le robot en fonction de ce qu'il perçoit.





On peut voir les robots dans de nombreux domaines, par exemple :

• **Les robots industriels :** soulagent les travailleurs de tâches répétitives ou dangereuses (ex : industrie nucléaire) ou les assistent dans des interventions qui requièrent un niveau de précision ou rapidité inaccessibles à un être humain.

• **Les robots d'intervention :** utilisés pour remplir des tâches dans des environnements où l'homme ne peut pas aller ou dangereux pour lui (ex : l'espace).

• **les robots domestiques :** exécutent des tâches autonomes dans la maison ou ont une relation directe avec les occupants (ex : robots-aspirateurs, les robots de télé-surveillance ...).

Les robots sont aussi utilisés pour représenter certains aspects des comportements humains animaux et humains, comme par exemple les mécanismes de la marche ou de l'apprentissage du langage. Cela peut aider les scientifiques à mieux comprendre ces comportements en testant avec les robots si les mécanismes qu'ils imaginent pour expliquer ces comportements peuvent fonctionner. Avec des robots, ils peuvent par exemple tester comment les informations captées par les sens (vue, ouïe) sont combinées pour produire des commandes motrices permettant au corps de se mouvoir.

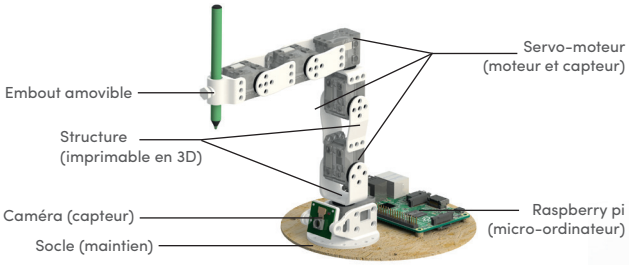
Source :
Oudeyer, P. Y. Où vont les robots ? *pyoudeyer.com/LettreMURS32.pdf*

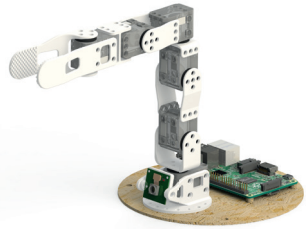
Les ancètres des robots sont les automates, qui sont des machines effectuant une tâche précise et toujours de la même manière. **Au contraire de l'automate,** un même robot se comportera de manière différente selon ce qui se passe dans son environnement.

Caractéristiques du robot Poppy Ergo Jr

Poppy Ergo Jr est un petit robot open-source de la famille Poppy. Ses pièces sont imprimées avec une imprimante 3D.

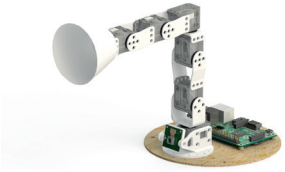
La plateforme robotique Poppy dont est issue le robot Poppy Ergo Jr, est open-source logiciel et matériel, c'est-à-dire que ses logiciels et ses plans sont **disponibles publiquement, librement utilisables et modifiables.** Grâce à cette approche, juridique et philosophique, des collaborations s'installent autour de projets et en permettant des améliorations et de nouveaux développements. C'est un moteur puissant de développement des nouvelles technologies.





Il est doté :

• **De capteurs** pour prendre des informations dans son environnement : sa caméra et ses **servo-moteurs** permettent de détecter ce qui se passe autour de lui ou sur lui (sa position, sa température etc.)



Un servomoteur est la combinaison d'un moteur et d'une carte électronique réalisant le contrôle précis (asservissement) du moteur dans une position, vitesse ou force donnée.

• **D'actionneurs** pour produire des actions : ses **servo-moteurs** permettent de bouger et ses leds émettent de de la lumière

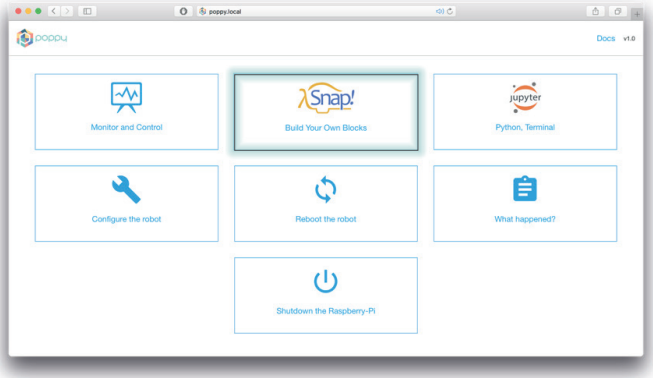
• **D'un micro-ordinateur :** connecté aux capteurs et aux actionneurs, il permet de **contrôler** le comportement du robot en exécutant un programme informatique qui contient les instructions nécessaires

Dans un robot, il y a un ordinateur, parfois tout petit. C'est le même type d'ordinateur qu'on utilise dans la vie quotidienne (ceux avec un clavier et une souris). Il permet de contrôler le robot, lui donner des instructions et d'enregistrer les données. Pour l'Ergo Jr, cet ordinateur s'appelle Raspberry Pi.

Commencer en Snap!

À faire avant de commencer : construire, brancher votre robot Poppy Ergo Jr, et se connecter sur sa page d'accueil.

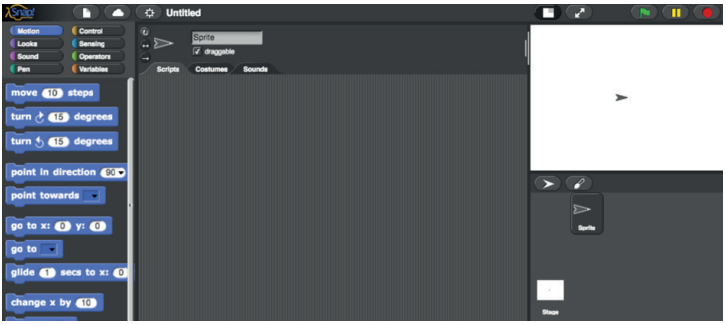
A Cliquez sur la case Snap! de la page d'accueil :



L'interface Snap! est découpée en trois parties :

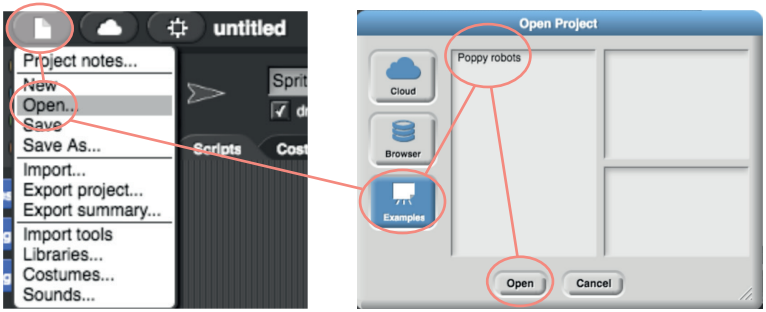
- **À gauche :** la liste des blocs (instructions) disponibles, rangés en différentes catégories.
- **Au centre :** la zone de scripts, où l'on dispose et assemble les blocs pour en faire des programmes.
- **À droite :** en haut une zone d'affichage contrôlable par les scripts, en bas les sprites, objets programmables pouvant évoluer dans la zone d'affichage.

Pour programmer Poppy Ergo Jr, vous utiliserez principalement les zones de gauche et du centre.



B Ouvrez le projet *Poppy robots* pour pouvoir accéder aux blocs spécifiques au robot Poppy :

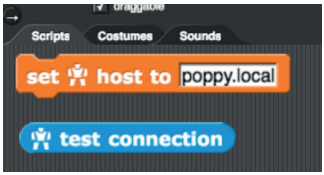
Pour cela cliquez sur : Fichier > Open > Examples > Poppy robots > Open



(Suite page suivante)

Astuce : Pour avoir des informations sur un bloc Snap! *cliquez droit* sur le bloc puis séléctionnez *help*.

Vous devriez voir apparaitre deux blocs dans la zone de scripts :



C Vérifiez que vous êtes bien connecté(e) au robot :

1. Écrivez le nom du robot ou l'adresse IP dans le bloc 

:

(Ici le nom du robot est *poppy*)

2. Cliquez sur le bloc 

pour vérifier que vous êtes bien connecté au robot :

Si un message d'erreur apparait :

Vérifiez que vous êtes bien branchés (reportez-vous à la documentation du robot, si besoin)

D Sauvegardez votre programme :

1. Vous pouvez l'enregistrer sur le disque dur de votre ordinateur en cliquant sur : *Files > Export > clic droit > Enregistrer sous*

2. Ou l'enregistrer dans le cloud, qui nécessite d'avoir un compte utilisateur :

Le stockage dans le cloud signifie que toutes les informations que vous consultez sont stockées sur des serveurs (ordinateurs) à divers endroits à travers le monde.

• Cliquez sur le menu *cloud* (nuage) dans la barre d'outils :

• Sélectionnez l'option *sign up* dans le menu, et suivez les instructions.

• Vérifiez votre email pour obtenir votre mot de passe initial.

Avant de contrôler Ergo Jr, vérifiez à chaque fois la bonne position du robot.

Vous êtes prêts pour commencer les activités !

Apprendre à programmer
Ergo Jr en Snap!

Partie 1 : *Contrôler Poppy Ergo Jr*

Vous allez apprendre à faire bouger le robot Poppy Ergo Jr en utilisant le langage de programmation Snap!. A la fin de la séance vous pourrez appliquer vos connaissances avec le défi **Ergo Jr joue au chamboule-tout.**

Un **langage de programmation** permet d'écrire un programme informatique, qui est une suite d'instructions à exécuter. Ceci permet de donner des comportements à un robot. Snap! est un langage de programmation avec lequel on assemble des *blocs d'instructions*. Un assemblage de blocs s'appelle un *script*.

Votre premier programme

Pour trouver des blocs dans Snap!, vous pouvez chercher :

• Par **couleur/catégorie** (chaque catégorie à une couleur) :

• Par **mots clés** (> Clic droit sur la partie de gauche > *find blocks*) et saisir :

- Un mot se trouvant dans le bloc souhaité (exemple : *when*)

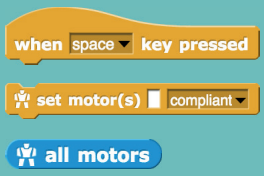
- Le mot clé *robot* afin de séléctionner seulement les blocs spécifiques au robot Ergo Jr

À vous de jouer !

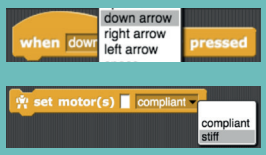
A Créez les deux scripts ci-dessous afin de pouvoir mettre Ergo Jr dans des positions spécifiques :

Pour cela :

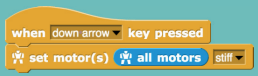
1. Sélectionnez et déposez les trois blocs dont vous avez besoin sur l'espace de travail central :



2. Cliquez sur les listes et choisissez les bonnes valeurs :

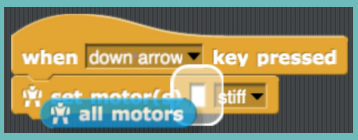


3. Assemblez les blocs ensemble :



4. Faites de même pour le deuxième script.

Pour emboiter des blocs : sélectionnez le bloc et déposez-le à l'endroit désiré avec la souris. La bordure blanche indique que les blocs vont s'emboiter.



Vous pouvez copier/coller les blocs et scripts : *Clic droit* puis *duplicate*

B Activez les deux scripts (une bordure blanche apparaît autour d'un script activé) :

1. Appuyez sur ⇩ sur votre clavier puis manipulez le robot. Que remarque-t-on ?

2. Appuyez sur ⇧ sur votre clavier puis manipulez le robot. Que remarque-t-on ?

C Qu'est-ce que le mode *compliant* du robot ? Et le mode *stiff* ?

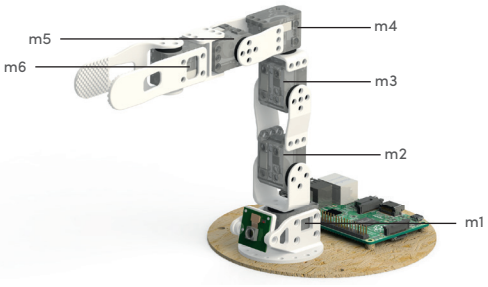
D Activez ces scripts et manipulez Ergo Jr pour le mettre dans différentes positions.

Des idées : donnez-lui l'air curieux, timide, content. À vous d'imaginer !

Faire bouger Ergo Jr à l'aide de ses moteurs

On utilise le bloc suivant pour faire bouger Ergo Jr, moteur par moteur :

Voici le schéma du robot avec le nom de chacun des moteurs :



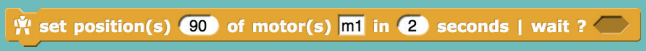
i Il est **nécessaire** d'activer les moteurs (**stiff**) du robot avant de pouvoir le **faire bouger** avec Snap!

À vous de jouer !

A **Assurez-vous que tous les moteurs soient bien activés (mode *stiff*)** et mettez tous les moteurs en position de base (qui correspond à la position où chaque moteur est à 0 degrés : aligné sur l'encoche) en cliquant sur le bloc suivant pour l'exécuter :

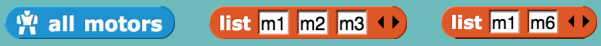
i Ici on utilise le bloc *set position(s)* : il accepte des valeurs de positions en degrés, il est conseillé de respecter un intervalle de [-90 ; 90], suivi du/des nom(s) du/des moteur(s), puis de la durée en secondes qu'il va mettre pour atteindre cette position. Concernant, la valeur *wait?* nous verrons son utilisation plus tard.

B Mettez le moteur *m1* dans la position 90 degrés en 2 secondes.



C

Cherchez les blocs ci-contre et exécutez-les chacun leur tour :



1. Quelles valeurs ces blocs renvoient-ils ?

i

Les blocs ont des formes différentes, chaque forme correspond à une catégorie spécifique.

• Les blocs de forme ovale (comme 

) sont appelés *reporter* : quand il sont exécutés, ils renvoient une valeur.

• Au sommet du script peut se trouver un bloc *Hat* (chapeau), qui indique quand le scénario doit être exécutés. Les noms de blocs *Hat* commencent généralement par le mot *When* (exemple : 

); un script n'a pas obligatoirement un bloc *Hat*, mais sans ce bloc, le script sera exécuté seulement si l'utilisateur clique sur le script lui-même.

• Les blocs *command* (comme 

) correspondent à une action.

2. Modifiez le bloc *set position* pour mettre le moteur *m1* et le moteur *m6* dans la position -30 degrés en 2 secondes.

D

En vous aidant des blocs que nous venons de découvrir, construisez deux programmes correspondant aux instructions ci-dessous :

1. Quand ⇨ sur votre clavier est pressée alors mettre tous les moteurs en position 0 degrés en 3 secondes.

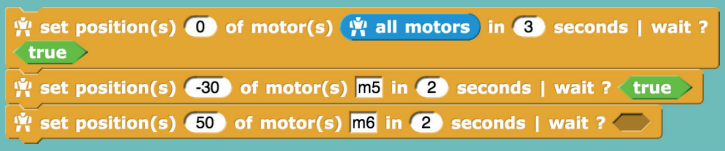
2. Quand ⇦ sur votre clavier est pressée alors mettre les moteurs m1 et m4 en position 60 degrés en 2 secondes.

Créer des mouvements

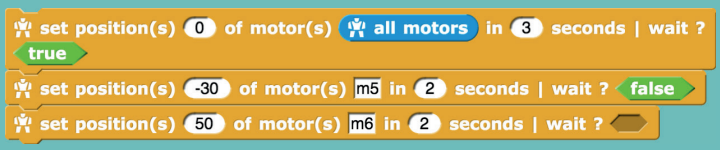
Maintenant, nous allons utiliser les moteurs pour créer des mouvements.

À vous de jouer !

A Exécutez le script ci-dessous et observez ce qui se passe :



B Remplacez le deuxième bloc `true` par le bloc `false`



1. Lorsqu'il y a deux blocs (ou plus) d'emboités, dans quel ordre s'effectuent les actions ?

2. Que se passe-t-il quand *wait* est égal à *true* ? Que se passe-t-il quand *wait* est égal à *false* ?

i Les lignes de code s'exécutent de façon quasi instantanées; et même si parfois la position demandée dans la ligne précédente n'a pas été atteinte. La partie *wait* permet d'attendre que le moteur ait atteint la position voulue avant d'exécuter la commande suivante.

C Avec les blocs que vous connaissez maintenant, faites jouer un mouvement à Ergo Jr signifiant "bonjour" quand on appuie sur la touche *b*.

Conseils:

• Commencez par un mouvement simple puis enrichissez-le au fur et à mesure.

• Choisissez les moteurs que vous souhaitez utiliser pour la création du mouvement.

• Faites jouer le mouvement choisi au robot (**en mode *compliant***) et observez les actions de chaque moteur.

• Vous pouvez vous aider du bloc 

pour connaitre la position d'un moteur ciblé, et ainsi noter la valeur pour la réutiliser ensuite.

• Programmez le mouvement moteur par moteur et testez à chaque fois le résultat de votre programme.

N'hésitez pas à créer d'autres mouvements !

Défi robotique : *Ergo-Jr joue au chamboule-tout*

Matériel:

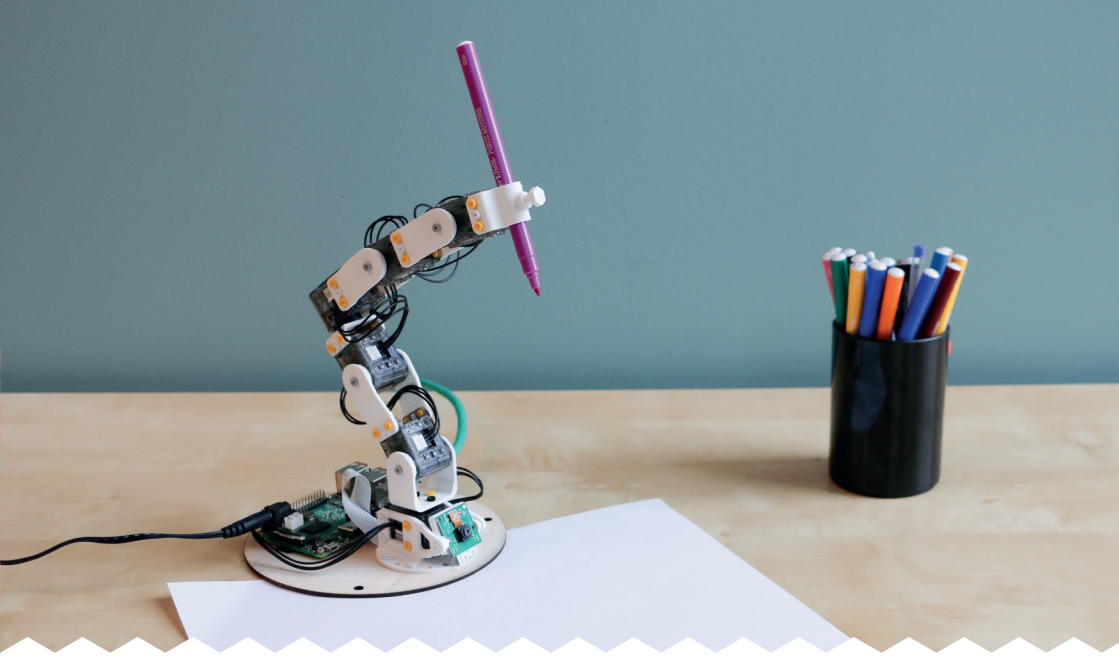
- Poppy Ergo Jr avec l'abat jour
- Une balle légère
- Des gobelets (carton ou plastique)

Objectif:

Contrôler la position et la vitesse des moteurs du robot pour lancer la balle et faire tomber le chamboule-tout.

Il y a de nombreuses manières possibles de lancer la balle. Combien pouvez-vous en trouver ?

Apprendre à programmer
Ergo Jr en Snap!

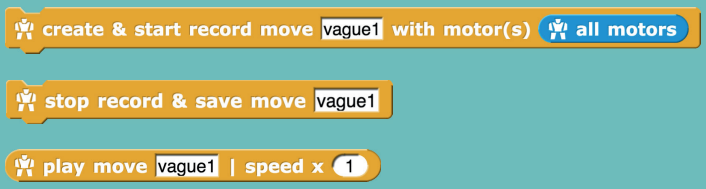


Partie 2 : *Programmer par démonstration*

Le robot Poppy Ergo Jr est capable de mesurer en temps réel la position de ses moteurs. Ainsi lorsqu'on les fait bouger manuellement, il peut **enregistrer les mouvements effectués pour les reproduire ulterieurement.**

À vous de jouer !

A Avec les trois blocs ci-dessous enregistrez un mouvement de vague puis jouez-le ensuite :



Pour cela :

1. Créez et exécutez le bloc *create & start* pour commencer l'enregistrement

2. Manipulez le robot pour créer un mouvement à reproduire

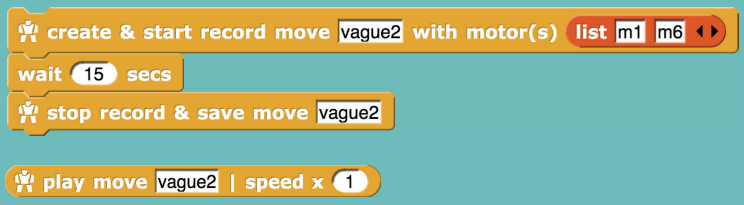
3. Arrêtez l'enregistrement avec le bloc *stop record*

4. Jouez le mouvement que vous avez créé avec le bloc *play move*

N'hésitez pas à refaire le mouvement jusqu'à ce qu'il vous convienne.

i Les mouvements sont sauvegardés dans un fichier qui se trouve dans l'ordinateur du robot Ergo Jr. Il faut **donner un nom unique à chaque mouvement** pour ne pas effacer le fichier précedent en ré-enregistrant un mouvement par dessus.

B Observez le programme ci-dessous sans le créer et essayez de deviner ce qu'il fait :



C Créez et testez le script pour vérifier.

À vous ! Enregistrez les mouvements de votre choix.

Pour aller plus loin...

De nombreuses options de lecture sont disponibles : expérimentez-les !

Que se passe-t-il dans les cas suivants :

A Modifiez dans le bloc 

la valeur de *speed* (avec un *nombre décimal ou non* allant de 0 à 4) ?

B Utilisez le bloc 

.

C Utilisez le bloc 

.

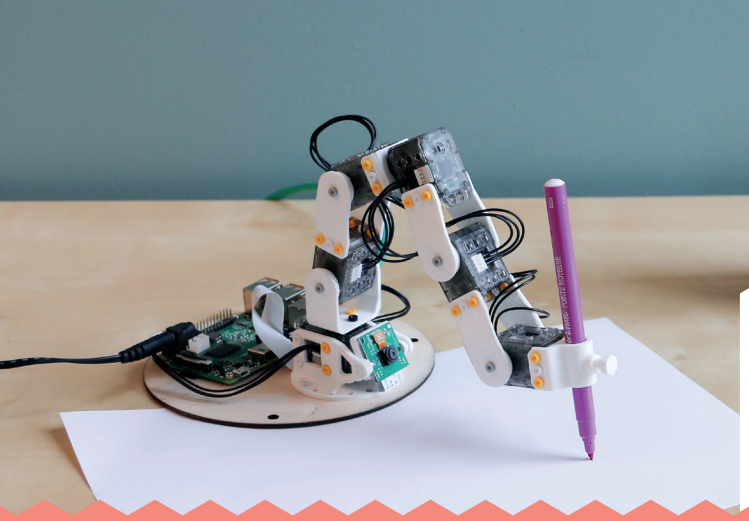
D Jouez trois mouvements différents avec le bloc 

.

E Jouez un mouvement qui a été enregistré avec le moteur m1 (*mouvement_m1*) et un deuxième mouvement qui a été enregistré avec les moteurs m5 et m6 (*mouvement_m5_m6*) avec le bloc 

.

i En robotique, les mesures de positions ne sont jamais parfaites. C'est le cas également sur le robot Ergo Jr, il est possible donc que le mouvement rejoué ne corresponde pas exactement à ce qui a été montré.



Défi robotique : *Ergo-Jr joue au «Dessiner c'est gagner !»*

Matériel:

- Poppy Ergo Jr avec le porte crayon
- Un feutre
- Une liste de mots. Exemples : soleil, marguerite, camion, escargot, chat, plage, etc. Une liste complète est disponible en annexe à la fin du livret.

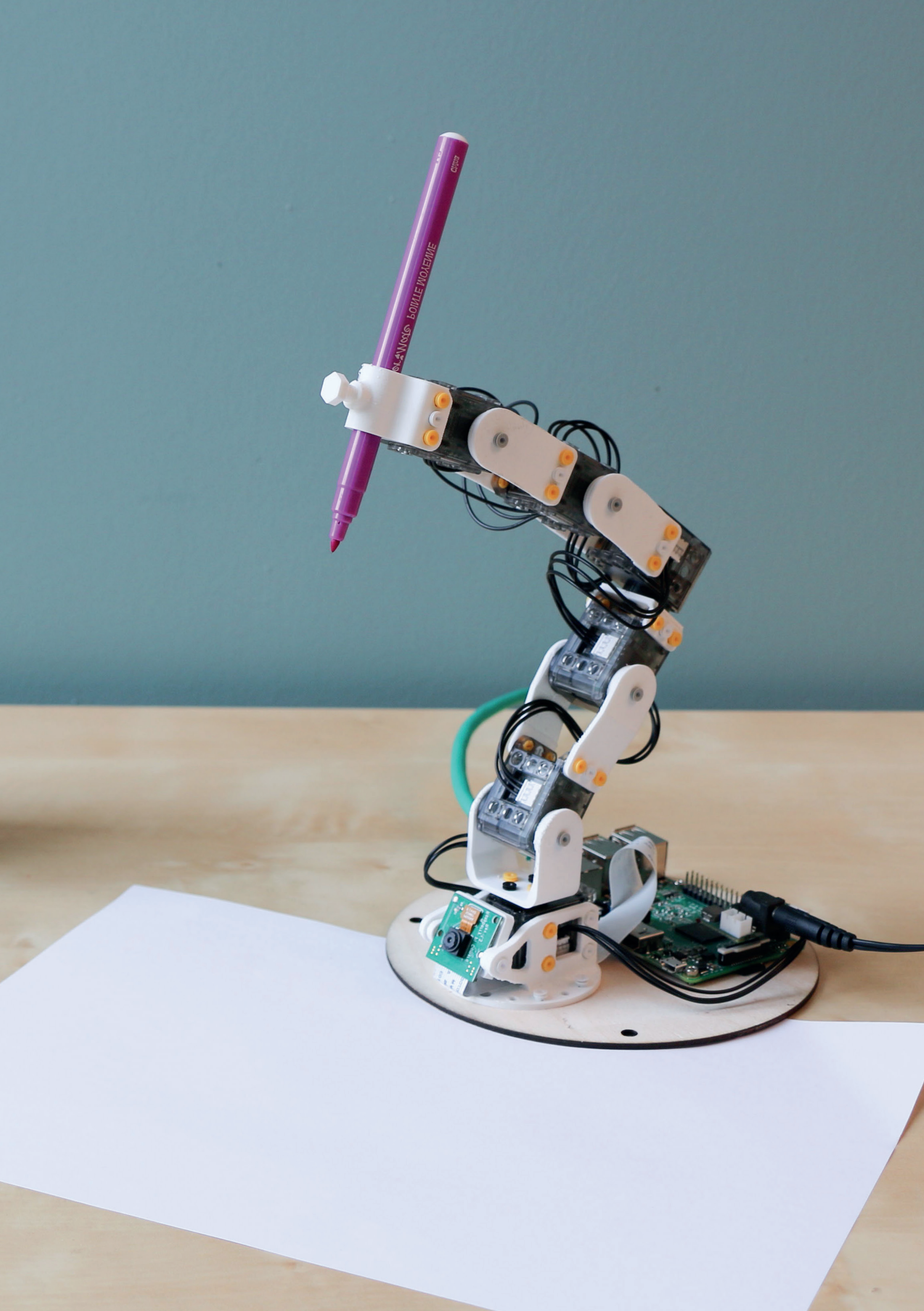
Objectif:

Phase de préparation :

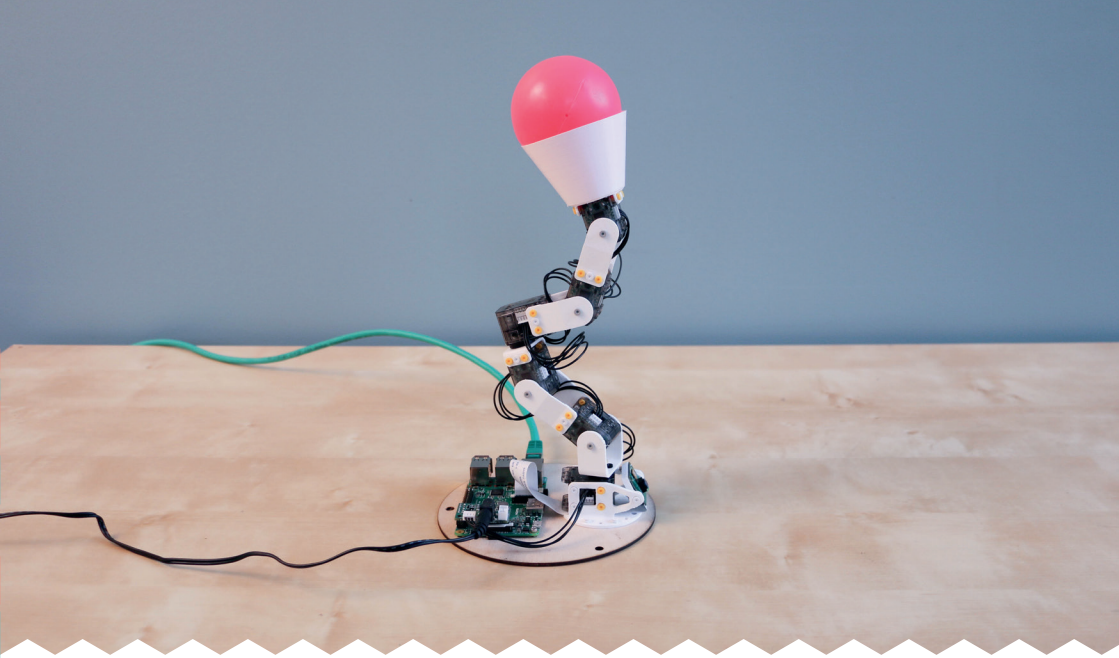
- Chaque équipe tire au sort des bouts de papier de la liste de mots
- En un temps limité (exemple : 5 min), enregistrez le maximum de dessins en programmant par démonstration

Phase de jeu :

- Chaque équipe fait deviner aux autres les mots illustrés en un temps limité (exemple : 40 s)
- Si le mot a été deviné :
  - La personne qui a deviné marque deux points
  - Chaque personne de l'équipe marque un point
- Si le mot n'a pas été deviné, personne ne marque de point.



Apprendre à programmer
Ergo Jr en Snap!



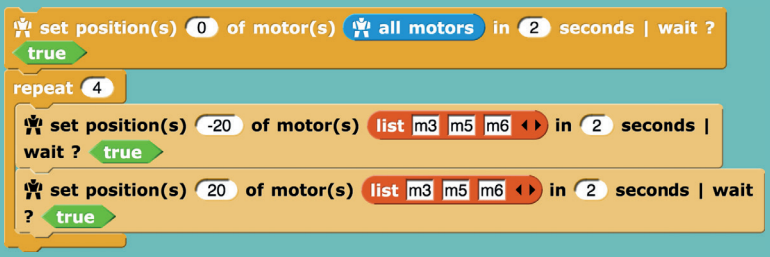
Partie 3 :
Utiliser la répétition

Vous allez découvrir à cette occasion un concept essentiel en informatique et en robotique : **les boucles** ! Vous allez faire danser Ergo Jr grâce au bloc en répétant plusieurs fois la même action.

On répète !

À vous de jouer !

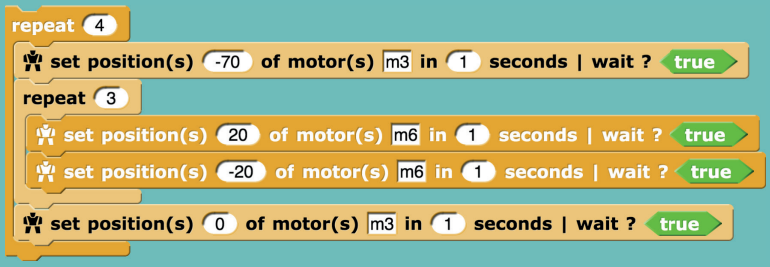
A Créez et exécutez le script ci-dessous et observez ce qui se passe :



1. Que se passe-t-il si on change la valeur 4 du bloc *repeat* par une autre ?

2. D'après vous, quels sont les différents intérêts du bloc *repeat* ?

B Observez le script ci-dessous sans le créer et dîtes ce qu'il fait :



1. Combien de fois le moteur *m3* va-t-il bouger ?

2. Combien de fois le moteur *m6* va-t-il bouger ?

C En utilisant le bloc 

, créez un mouvement pour que le robot vous fasse un petit salut pour vous féliciter.

Grâce aux boucles, ici le bloc Repeat, les scripts sont plus courts, apportent souvent plus de clarté et permettent également de faire des choses plus élaborées.

Action !

À vous de jouer !

A Cherchez le bloc 

et exécutez-le plusieurs fois pour l'essayer.

1. Quelle valeur renvoie-t-il ?

2. Modifiez le bloc pour qu'il renvoie une valeur comprise entre [-80 ; 80]

B Maintenant, utilisez ce que vous venez d'apprendre pour faire danser Ergo Jr de manière aléatoire.

Pour cela :

1. Mettez le moteur *m1* à une position random comprise entre -80 et 80.

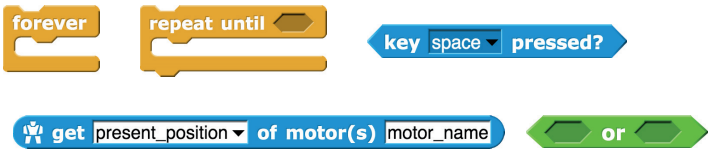
2. Faites la même chose avec le moteur *m3* en choisissant un intervalle de positions qui vous semble convenable (si besoin, aidez-vous du bloc 

).

3. En utilisant un (ou plusieurs) bloc *repeat*, faites bouger **chaque moteur les uns après les autres** (choisissez bien les intervalles).

Pour aller plus loin...

Expérimentez les blocs ci-dessous et créez des scripts les utilisant :



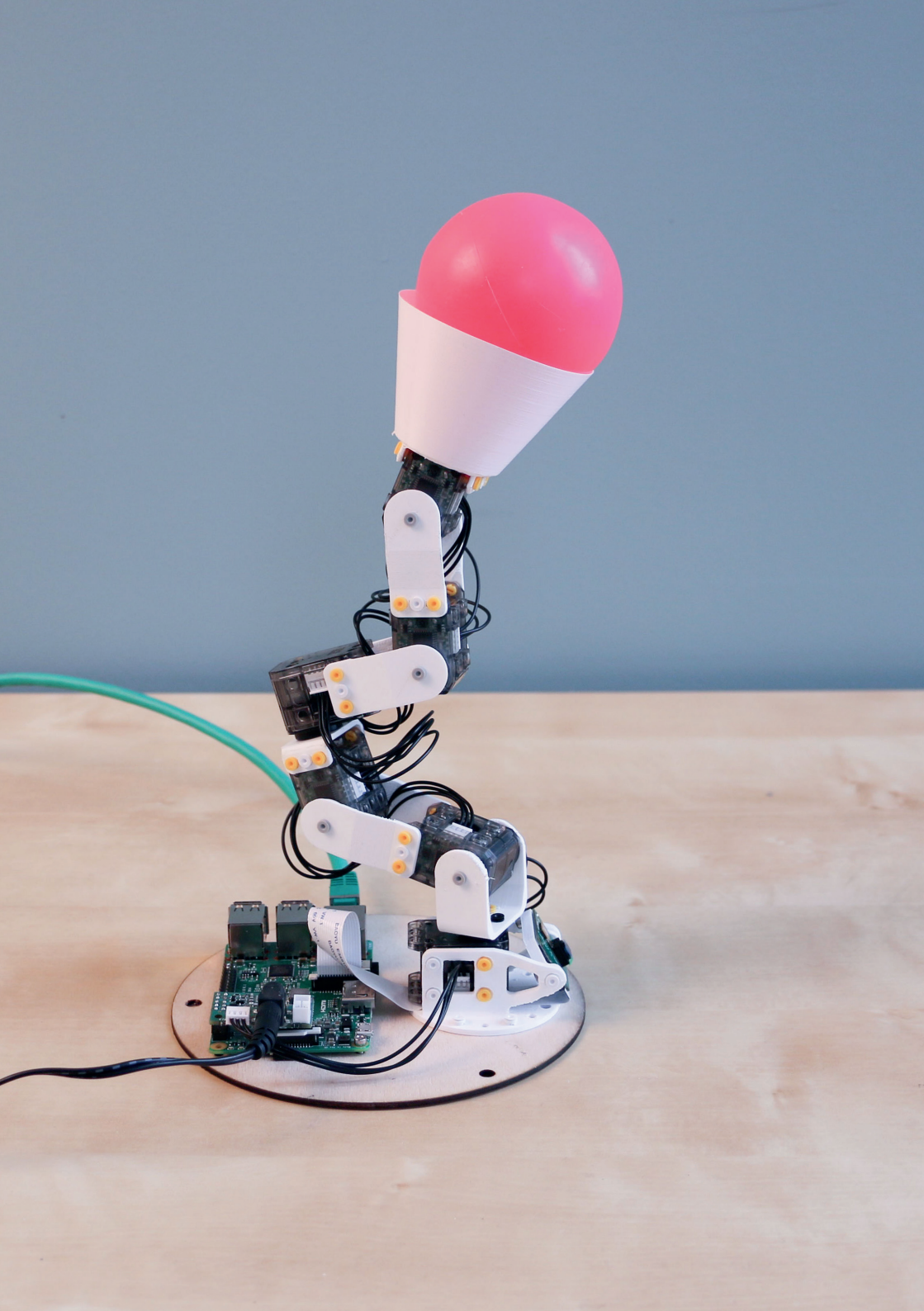
Défi robotique : *Ergo Jr l'otarie fait son cirque !*

Matériel:

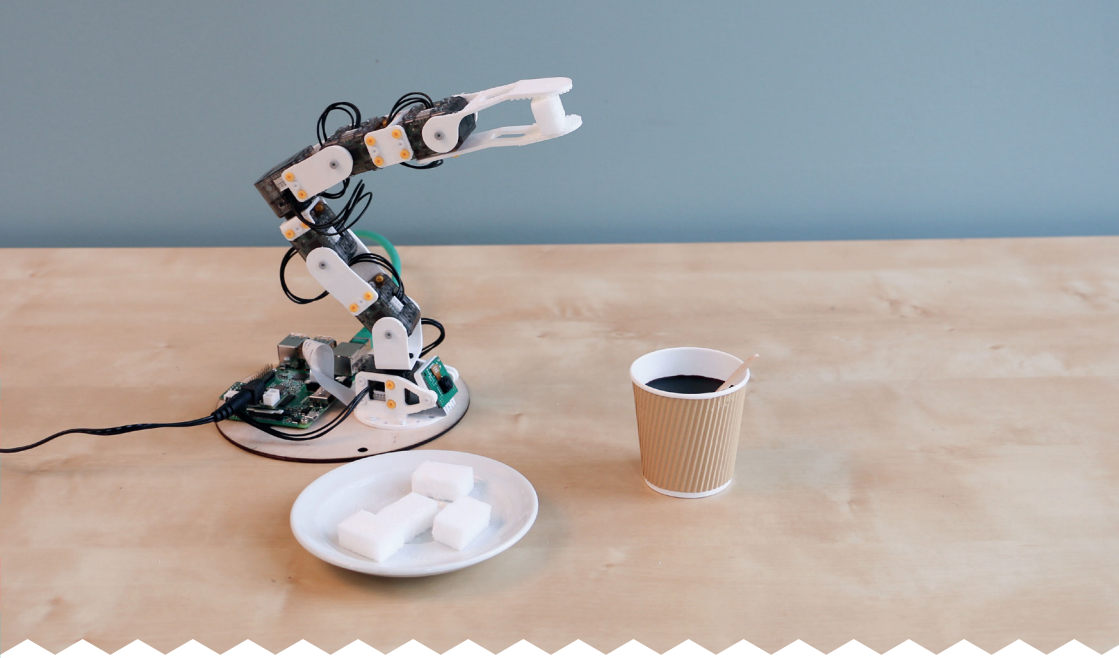
- Poppy Ergo Jr avec l'abat-jour
- Une balle légère

Objectif:

- Mettez la balle dans l'abat-jour et faites danser Poppy Ergo Jr de manière aléatoire avec répétition de mouvements.
Utilisez tous les moteurs, attention à ne pas faire tomber la balle !



Apprendre à programmer
Ergo Jr en Snap!



Partie 4 : *Créer son propre bloc Snap!*

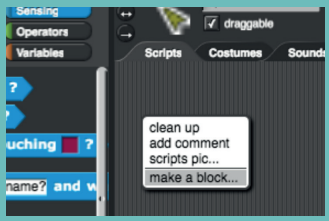
Vous allez apprendre ici à créer vos propres blocs pour pouvoir stocker, réutiliser et modifier les comportements que vous avez créés.

Créer un bloc

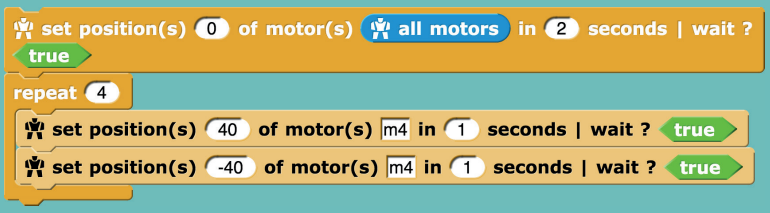
À vous de jouer !

i En Snap! chaque bloc a besoin d'une couleur, d'un titre, d'une catégorie (forme), et d'un script qui définit son comportement.

A Cliquez droit sur un endroit vide de la zone de script et sélectionnez *make a block...* :



B Créez le bloc **kidinon** qui lancera le script ci-dessous :



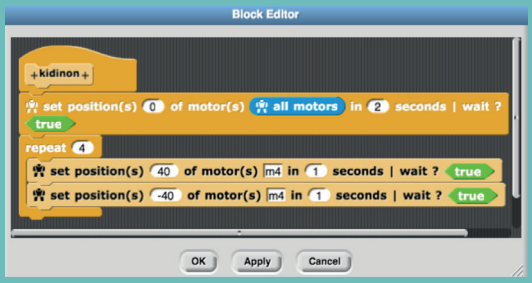
Pour cela :

1. Choisissez la catégorie *Control* de couleur jaune pour ranger votre nouveau bloc.

2. Donnez au bloc un nom qui décrit l'action du script.

3. Séléctionnez la catégorie *Command* (car nous voulons un bloc qui agit)

4. Construisez dans la zone *block editor* le script de votre nouveau bloc puis validez. *(page ci-contre)*



Vous pouvez rechercher votre bloc (par catégorie ou par nom) et l'utiliser de la même façon que les autres.

Ajouter une entrée *(input)*

i On utilise ***une entrée*** pour demander à l'utilisateur **une information précise ou pour indiquer une action;** voici des blocs avec des entrées :

À vous de jouer !

A Ajoutez une entrée, à votre bloc, pour permettre à l'utilisateur de modifier le nombre de répétition du mouvement *kidinon* : 

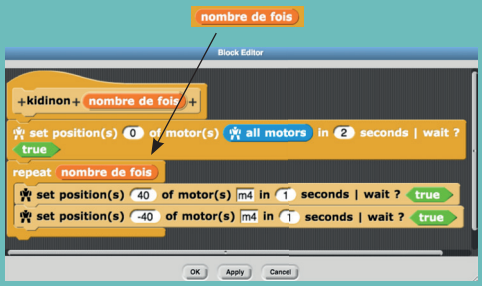
1. Cliquez droit sur le bloc et sélectionnez *edit* :

2. Cliquez sur le + à droite :

3. Laissez le bouton *input name* sélectionné et écrivez *nombre de fois* :

(suite page suivante)

4. Vous pouvez maintenant déposer la variable *nombre de fois* sur l'entrée du bloc *repeat* :



> Une variable est une zone mémoire désignée par un nom qui peut contenir des valeurs d'un certain type (voir la partie 7).

5. Validez et testez votre bloc `kidinon`.

B Ajoutez le mot fois : `kidinon fois`

Cette fois, sélectionnez *title text* :

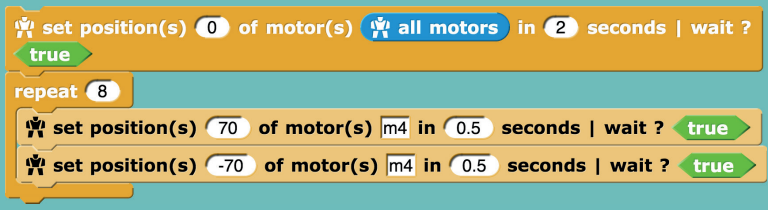
C Ajoutez une entrée permettant à l'utilisateur de modifier la *durée* du mouvement : `kidinon fois en seconde(s)`

D A l'aide d'un bloc de la catégorie operators (vert) : ajoutez une entrée *amplitude* pour permettre à l'utilisateur de modifier les positions des moteurs durant l'exécution du mouvement *Kidinon* :

`kidinon fois en seconde(s) avec une amplitude de -/+ degrés`

E Modifiez les données du bloc que vous venez de créer pour reproduire l'équivalent de ces deux mouvements :

```
set position(s) 0 of motor(s) all motors in 2 seconds | wait ?
true
repeat 4
    set position(s) 40 of motor(s) m4 in 1 seconds | wait ? true
    set position(s) -40 of motor(s) m4 in 1 seconds | wait ? true
```



Pour aller plus loin...

i Il est possible de choisir le type de données accepté pour chaque *entrée*. Par exemple pour le bloc :

Il est seulement possible de saisir des chiffres dans les zones de saisie de forme ovale.

A Cliquez sur la flèche de droite de la zone d'édition des entrées et explorez les options :

1. Modifiez la zone de saisie et acceptez seulement des nombres

2. Indiquez une valeur par défaut

3. Clic droit sur un type d'entrée et sélectionnez *options*: à quoi cela sert-il ?

B Documentez votre bloc (*clic droit* dans l'éditeur du bloc > *comment*) : pour que le texte s'affiche dans l'aide du bloc.

Apprendre à programmer
Ergo Jr en Snap!

Partie 5 :
Si... alors... !

Un branchement conditionnel de type Si... Alors... permet de faire exécuter des instructions selon si une condition donnée est vraie ou non.

Condition vraie, condition fausse

Les blocs *reporter* (blocs arrondis) peuvent **renvoyer différentes valeurs** (par exemple le bloc ( ) + ( ) renvoie un nombre).

Les blocs *predicates* (blocs de forme pointue comme celui ci : < >) **renvoient toujours une valeur** true ou false (on parle de valeur ***booléenne***).

Dans une expression logique (dite aussi booléenne), il n'y a que deux alternatives ; soit *Vrai (True)* ou soit *Faux (False)*. **Par exemple :**

On peut les combiner :

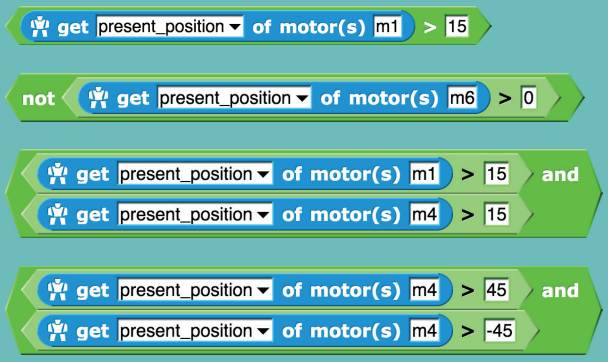
L'affirmation "5 + 2 = 7" est vrai et l'affirmation "10 + 2 = 7" est fausse donc l'affirmation complète est fausse.

L'affirmation "3 + 2 = 5" est vrai et l'affirmation "10 + 2 = 12" est vraie donc l'affirmation complète est vraie.

À vous de jouer !

A Dîtes, sans utiliser Snap!, si les expressions logiques ci-dessous sont vraies ou fausses puis construisez les blocs pour vérifier vos réponses.

B Pour chacune des expressions logiques ci-dessous, manipulez votre robot pour faire en sorte que les expressions soient vraies (true) puis faites en sorte qu'elles deviennent fausses (false).



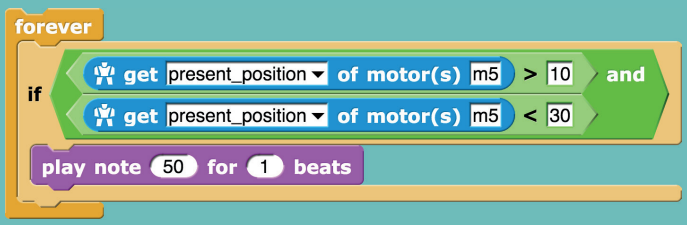
N'hésitez pas à en créer d'autres !

Transformez votre robot en instrument de musique !

Pour jouer une musique en fonction de différentes positions du robot, nous allons créer un script utilisant le concept de condition.

À vous de jouer !

A Créez le script ci-dessous et executez-le.



(suite page suivante)

1. Changez la position du moteur m5 (manuellement ou avec Snap!) puis observez ce qui se passe.

2. Modifiez le script pour faire en sorte de jouer une note de “60 for 0,5 beats” si le moteur m4 se situe dans une position entre 0 et 60 degrès.

B En utilisant le bloc 

créez un script pour faire en sorte de jouer une note de musique si tous les moteurs se situent dans une position entre -5 et 5 degrés.

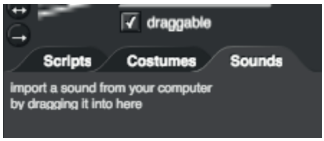
i Il est aussi possible d’emboiter des blocs 

pour créer des conditions imbriquées.

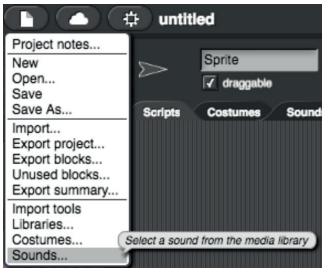
Pour aller plus loin...

Explorez les blocs se trouvant dans la partie “Sound” avec ce que vous venez d’apprendre précedemment.

i Pour utiliser un fichier son enregistré dans votre ordinateur, faîtes : “glisser/déposer” d’un fichier de musique dans la partie “Sounds”.



Il y aussi des sons proposés par défaut qu’il faut ouvrir en cliquant sur Fichier > Sounds.



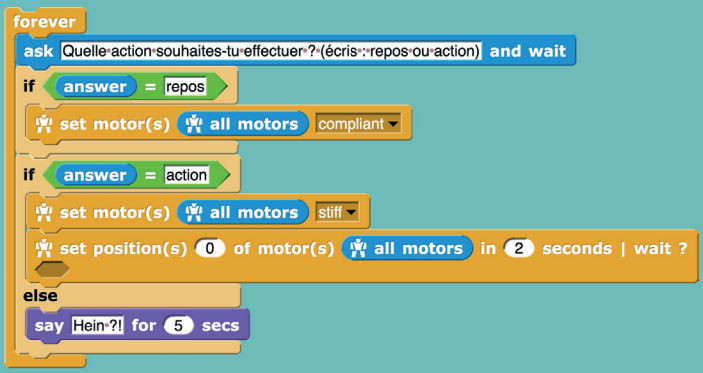
Transformez votre robot en instrument de musique !

Nous allons maintenant utiliser un autre exemple pour comprendre le bloc 

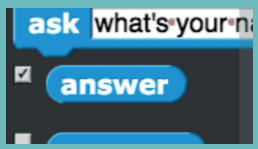
.

À vous de jouer !

A Créez le script ci-dessous et éxecutez-le :

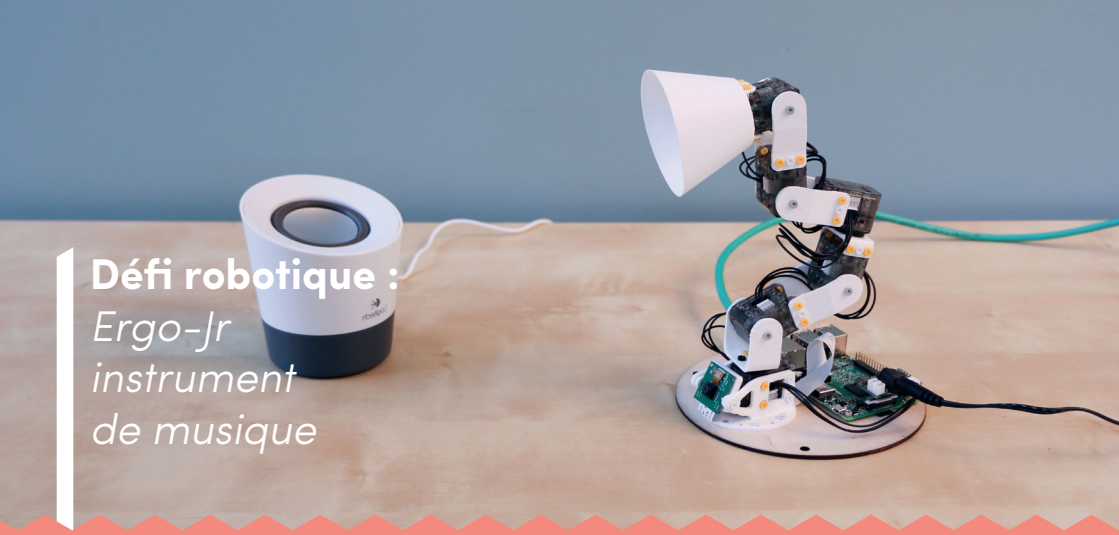


i Vous pouvez cocher la case *answer* qui se trouve à droite dans la catégorie *sensing* (bleu clair) pour voir apparaitre, à gauche, en temps réel la valeur du bloc *answer*.



B Que fait le script ? Dans quel cas renvoie-t-il *Hein ?!* ?

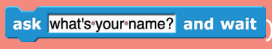
C Dans le script ci-dessus, la réponse de l'utilisateur (*repos* ou *action*) peut provoquer deux actions différentes. Modifiez le script pour compléter la liste des actions possibles (par exemple : bonjour, danse, High Five (tape m'en cinq) etc.) et créez les comportements qui vont avec.



Défi robotique :
Ergo-Jr instrument de musique

Objectif:

Utilisez Poppy Ergo comme un instrument de musique; en fonction de la position des moteurs il peut jouer des musiques différentes. Comme contraintes, vous pouvez :

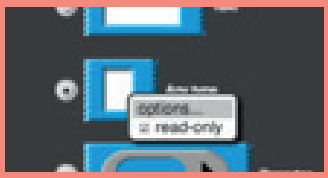
• Créez de l'interaction avec l'utilisateur (bloc 

)

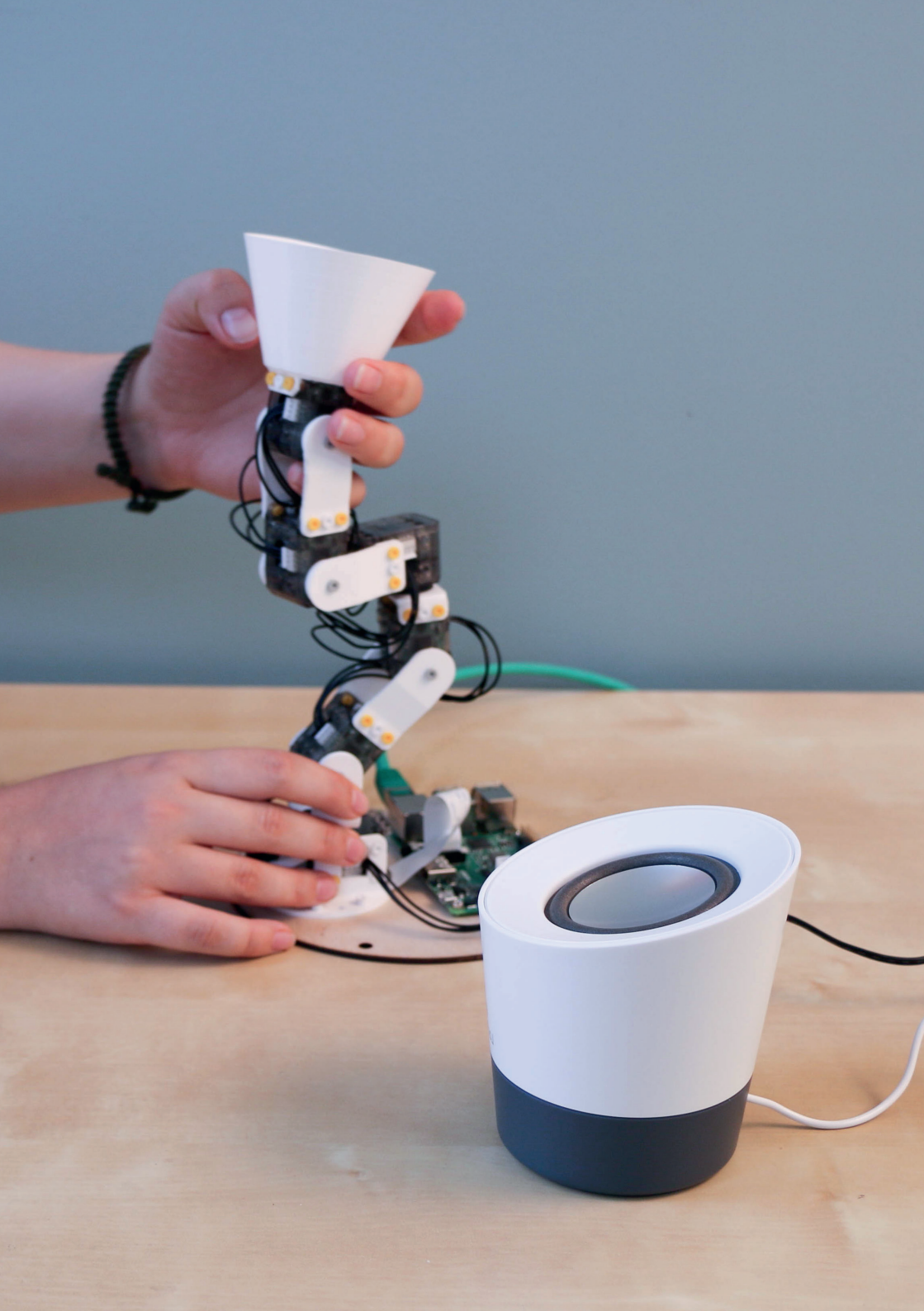
• Créez un bloc de musique où l'on peut sélectionner :
- la note de musique : Do (C), Ré (D), Mi (E), Fa (f), Sol (g), La (A), Si (B), Do (C)

i Pour créer des options par défaut, comme pour ce bloc :

Il faut éditer la zone de saisie puis cliquez droit sur un type d'entrée et sélectionnez *options*



- La durée (noire, blanche etc.)
- L'octave (optionnel)



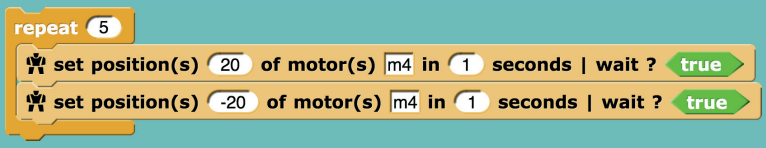
Apprendre à programmer
Ergo Jr en Snap!

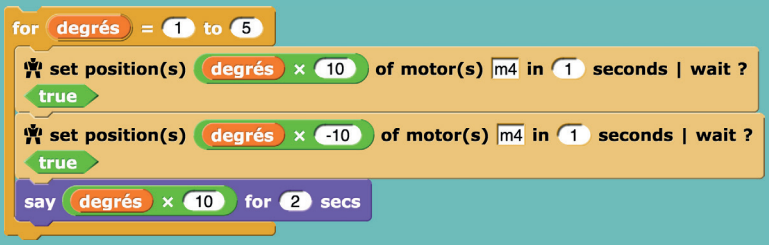
Partie 6 :
Le bloc **For**

Le bloc est un cas particulier des boucles que nous avons vues en partie 3 (utiliser la répétition). Il est très pratique pour répéter des instructions utilisant une valeur qui évolue.

À vous de jouer !

A Créez et exécutez les deux scripts ci-dessous.

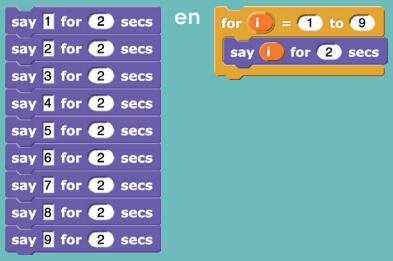




1. Quelle différence principale constate-t-on entre les deux scripts ?

2. Quelle est la valeur de la variable *degrés* pour chaque étape de la boucle ?

Le bloc *For* permet de simplifier un long script. Tel que :



B Modifiez le *bloc for* du second script : remplacez *1 to 5* par *5 to 1*. Que se passe-t-il ?

C Modifiez le script pour que le moteur *m4* change de position par pas de 20 degrés : 20 et -20 puis 40 et -40 puis 60 et -60 et enfin 80 et -80.

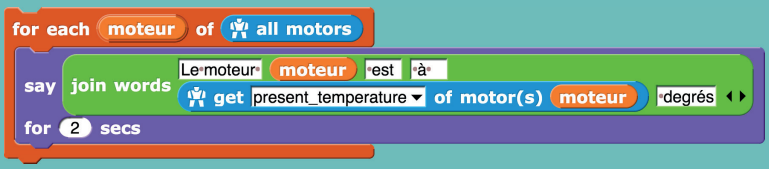
D Complétez avec ce nouveau mouvement, toujours par pas de 20 degrés : 80 et -80 puis 60 et -60 etc jusqu'à 10 et -10.

Un peu plus compliqué : le bloc 

permet de faire une action sur chaque élément (item) d'une liste.

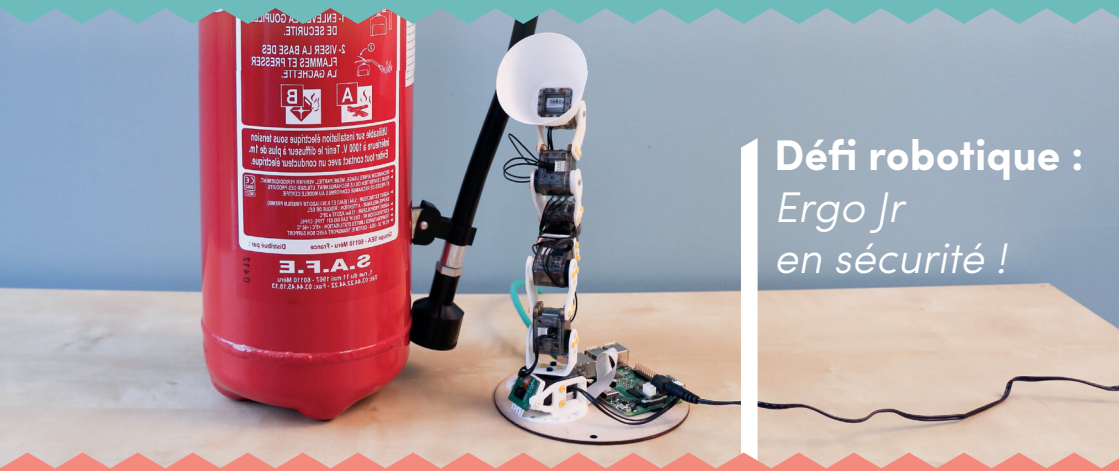
À vous de jouer !

Observez le script ci-dessous sans le créer et essayez de deviner ce qu'il fait :



1. Créez et testez le script pour vérifier votre intuition précédente.

2. Modifiez le script pour remplacer la liste **all motors** par une liste comprenant les moteurs m2, m3, m5 et m6.



Défi robotique : *Ergo Jr en sécurité !*

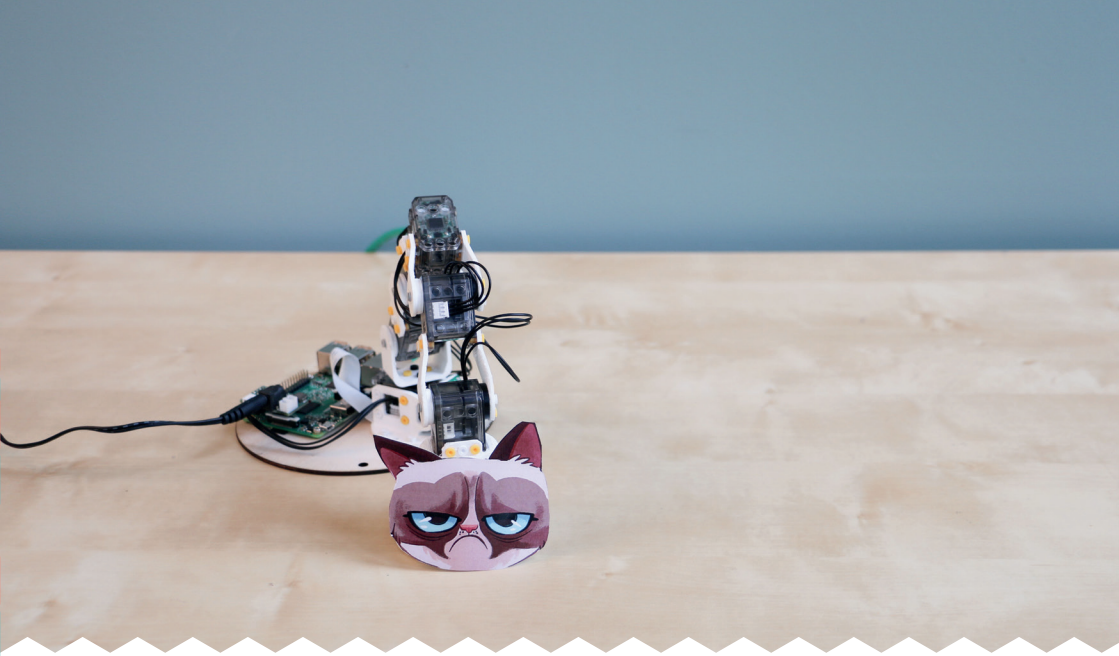
i Selon la position du robot et des comportements qu'il effectue, la température des moteurs peut augmenter. Lorsque la température d'un moteur est trop élevé, celui-ci est en erreur (led clignotante rouge) et peut même être endommagé. Le bloc **get present_temperature of motor(s) m1** permet de connaitre la température du moteur m1.

Objectif:

- Créez en Snap! l'alarme de votre choix (un son, un comportement etc.) qui se déclenche quand la température d'un moteur est trop élevé, avant qu'il se mette en sécurité (led clignotante rouge).

- Faites des essais pour estimer ce qu'est une température de déclenchement raisonnable pour votre Poppy Ergo Jr.

Apprendre à programmer
Ergo Jr en Snap!



Partie 7 :
Les variables

Vous avez :

Créé une variable comme entrée à un bloc,

Utilisé la variable **answer** pour stocker et réutiliser la réponse de l'utilisateur,

Manipulé la variable que le bloc vous donne,

Il est possible de créer d'autres variables à intégrer dans votre script avec le bloc

À vous de jouer !

A Créez, nommez une variable puis donnez lui une valeur :

Pour cela :

1. Créez une variable avec le bloc 

2. Nommez la variable en cliquant sur 

3. Donnez une valeur à la variable en utilisant le bloc 

.
Le menu déroulant permet de choisir la variable.

B Créez le script ci-dessous et exécutez-le plusieurs fois pour l'essayer :

i Vous pouvez utiliser le bloc 

pour afficher la valeur d'une variable.

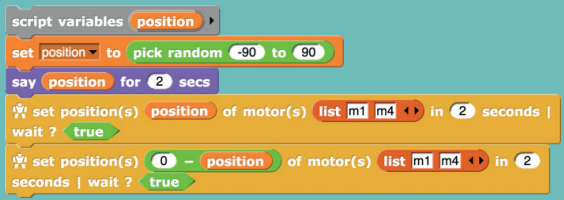
C Observez les deux scripts ci-dessous, essayez de deviner ce qui les différencie et vérifiez-le en les exécutant à tour de rôle.

En programmation, on a besoin de stocker des valeurs. Il peut s'agir, par exemple, de données fournies par l'utilisateur (frappes au clavier) ou de résultats obtenus par un programme. On dit qu'elle est « variable » car la valeur peut changer.

Utilisons maintenant la valeur générée par la variable du script sur votre robot.

À vous de jouer !

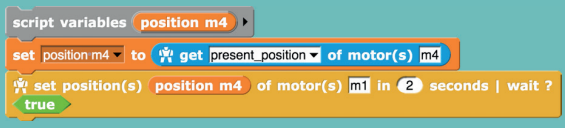
A Observez le script ci-dessous et essayez de deviner ce qu'il fait :



1. Créez-le et testez-le pour vérifier.

2. Modifiez le script et ajoutez une variable *durée* et donnée lui la valeur *1.5*

B Créez et exécutez le script ci-dessous et observez ce qu'il fait :

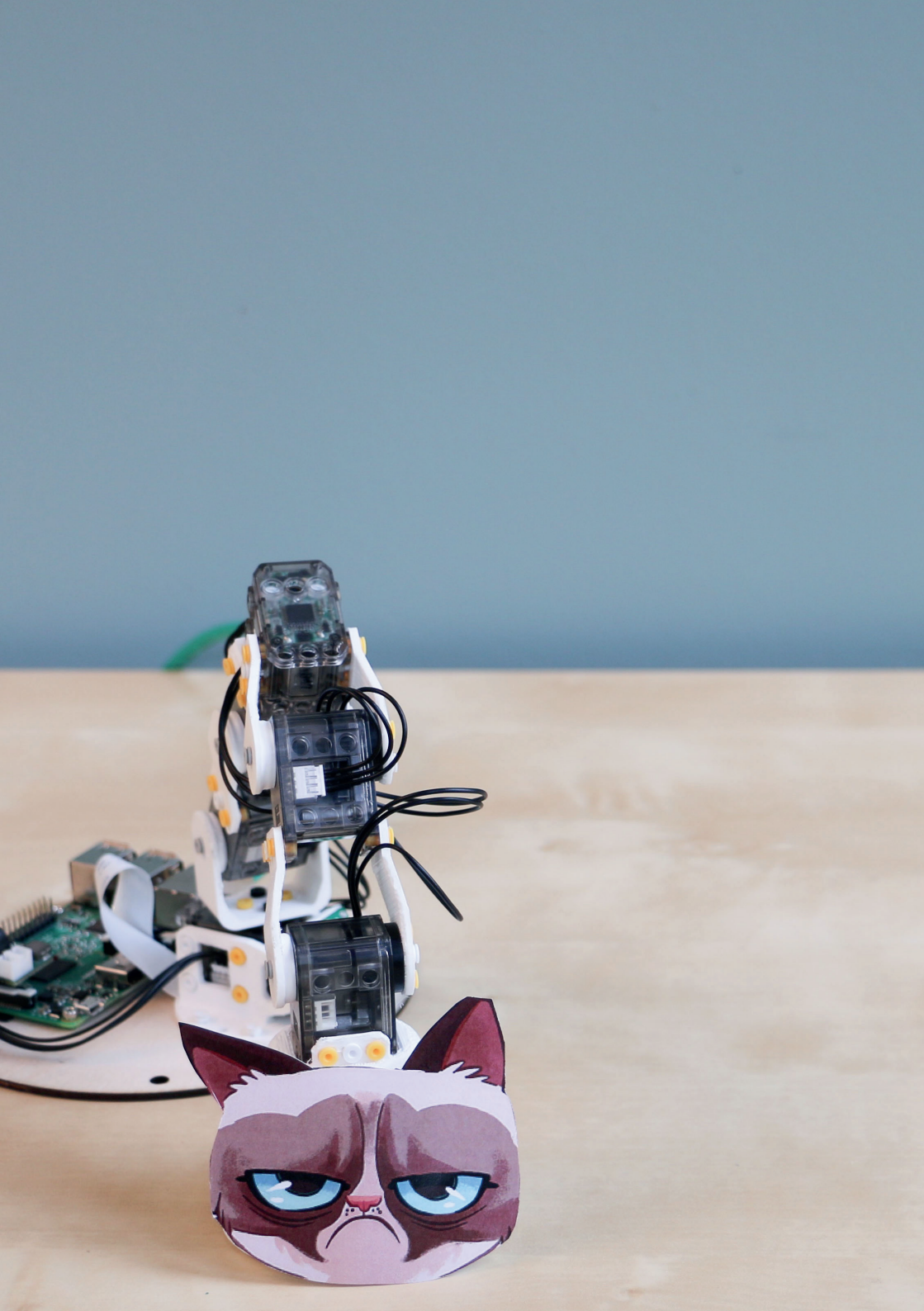


C Modifiez le script pour qu'il donne aussi au moteur *m6* la même position que le bloc *m5*.

Défi robotique : *Ergo-Jr est grincheux !*

Objectif:

Créez un programme qui détecte la position du robot Ergo Jr et qui le remet dans sa position initiale dès qu'une personne le manipule pour changer sa position.



Idées d'activités

À chaque idée présentée ici correspond un sujet dans le forum du livret (forum.poppy-project.org : catégorie Education), avec des astuces, des conseils, des échanges avec d'autres Ergoteurs !

Facile :

* *Ergo Jr en scène*

Créer un clavier-piano en associant une note à une touche du clavier et à une position d'un moteur. Ainsi, en appuyant sur les touches du clavier, on fait jouer à la fois des sons à l'ordinateur et des mouvements à l'Ergo Jr.

* *Ergo Jr range des conteneurs*

Programmer Ergo Jr de façon à ce qu'il saisisse et empile des boites (conteneurs) dans une zone spécifique, en fonction du QR code affiché sur celles-ci.
(le bloc 

permet de séléctionner un QR code : les codes à sont à la fin du livret).

* *Ergo Jr fait l'intéressant !*

Créer des positions et décorez/déguisez l'Ergo Jr pour illustrer des phrases en lien avec des sentiments ou des comportement puis en faire des gifs/ vidéos/images.
Par exemple : "quand j'ai peur... je me cache", "quand je suis heureux je fais le clown".

Variantes :
• Faire intervenir des QR codes avec des images qui représente, faire un gif/ vidéo
• Cela peut également se faire avec des comportements de la vie quotidienne : « quand un client est en colère... je reste calme et je souris ! ».

* *Ergo Jr fait son cinéma !*

Utiliser Ergo Jr pour contrôler des objets virtuels disponibles sur la scène de Snap!. Par exemple, programmer Ergo Jr pour qu'il fasse déplacer un lutin ou dessiner quelque chose sur la scène en le manipulant.

* *Ergo Jr fait l'explorateur !*

Utiliser les moteurs de la base en mode moteur, et ceux du bout en mode capteur (en les rendant compliant). On peut ainsi réaliser des missions de reconnaissance où l'on doit programmer le robot afin qu'il explore autour de lui et dès qu'il est touché dans une direction il focalise son exploration dans cette direction, ou dès qu'on le touche il fait un mouvement rapide de retrait, etc ...

* *Ergo Jr Sumo*

Trouver et tester des stratégies pour pousser un autre robot adverse ou un objet en dehors d'une zone.

** *Ergo Jr Garçon de café*

Construire un programme pour attraper un sucre et le mettre dans une tasse de café. Contrainte :

1. On souhaite donner au robot l'ordre de servir vers la gauche ou vers la droite par simple appui sur l'une des touches -> ou <-.
2. Un seul sucre doit être déposé dans la tasse. (le bloc 

permet de connaitre la force dévéloppée au moment t)

Variante :
Chercher et attraper un objet. Si l'objet est présent, Ergo Jr le déplace ailleurs, sinon il reprend sa position initiale.

** *Ergo Jr chamboule... tout !*

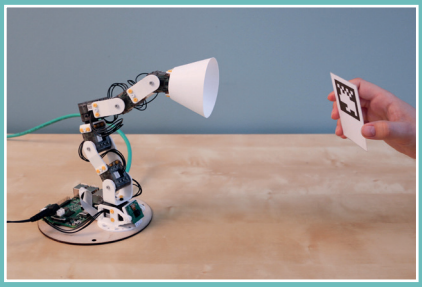
Contrôler la position et la vitesse des moteurs du robot pour lancer la balle et faire tomber le chamboule-tout (des gobelets les uns sur les autres).

Variantes :
• Lancer la balle dans un panier de basket ou une corbeille.

• Pour enrichir votre programme, vous pouvez associer des touches du clavier à des réglages (mise en position de chaque moteur, vitesse etc.)

• Créez un programme avec Snap! pour enregistrer le score (et le nombre d'essais). Ces résultats peuvent également être utilisés pour entraîner des comportements différents d'Ergo Jr (content, triste, ...).

• Des compétitions entre plusieurs Ergo Jr peuvent être organisées.



** *Ergo Jr invente un langage*

Inventer un langage de programmation basé sur l'utilisation de QR Codes. Pour cela, enregistrez un mouvement pour chaque QR code, programmez un bloc pour qu'Ergo Jr exécute ce mouvement si sa caméra détecte le QR Code associé. Faites de même avec plusieurs QR Codes. Ainsi, on peut faire jouer des chorégraphies au robot en montrant différents QR Codes à la suite pour enchaîner les mouvements.

Difficile :

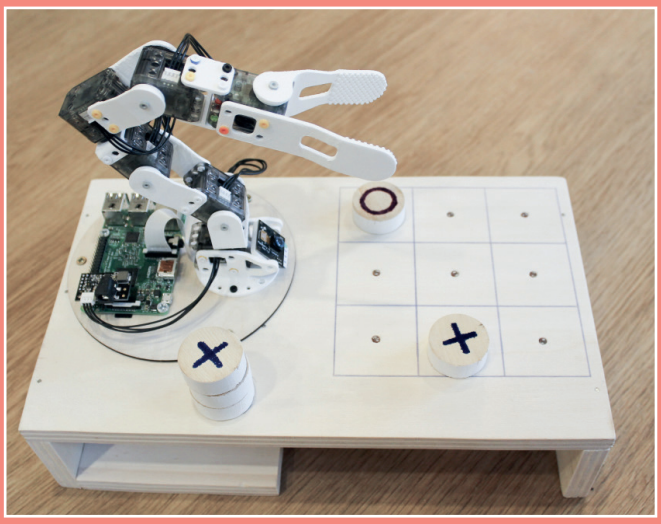


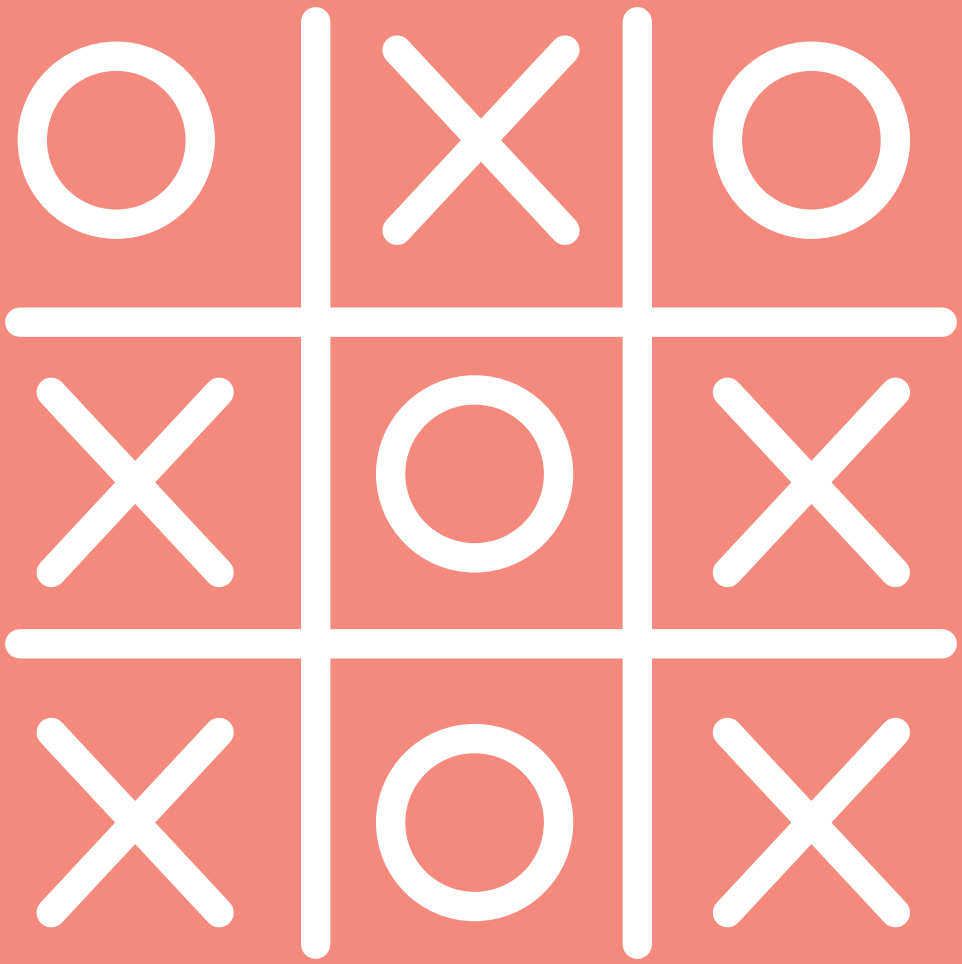
Photo par Gilles Lassus

*** Ergo Jr joue à Tic-Tac-Toe

Le jeu consiste à placer à tour de rôle une croix pour un joueur, un cercle pour l'autre. Pour gagner, un joueur doit aligner trois symboles identiques sur une même ligne, une même colonne ou une même diagonale.

Pour faire jouer Ergo Jr, on peut le programmer par démonstration afin qu'il sache atteindre chacune des 9 cases prévues pour le jeu et dessiner la croix (ou le cercle, selon le choix effectué). Il s'agit ensuite de le faire jouer par action sur une touche, avec un algorithme pour essayer de gagner.

Variante : les symboles sont sur des jetons empilés qu'Ergo Jr vient chercher pour les poser où il le souhaite sur le tapis de jeu.



Annexes

Ergo Jr • Liste de mots *Dessiner c'est gagner*

Facile :

un soleil
une lune
une marguerite
une enveloppe
une étoile
une main
une cerise
un nuage
un chapeau
un fromage
une bouteille
un escargot
un anneau
un ballon
une banane
une araignée
une glace
un chateau
une casserole
un papillon
une voiture
une maison

Moyen :

un camion
une pyramide
un escargot
un vase
une chaise
un avion
un bateau
un ours
un sac
une barbe
une clé
un visage
une chaussette
un lapin
une fourchette
un panier
de l'argent
une feuille
une montagne
un oeil
une moustache
un café
la pluie
du pain
un train

Difficile :

un chat
une plage
un coquillage
une horloge
une chaussure
un telephone
un immeuble
une vache
un requin
un ordinateur
une moto
des couettes
un oeuf sur la plat
une renne
un roi
un vélo
une poule
un cheval
un lion

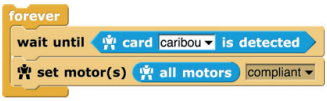
Annexes

QR codes à imprimer

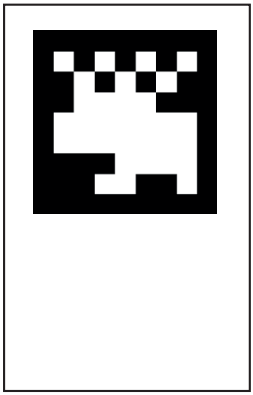
Le bloc 

permet de déclencher une action si la carte sélectionnée est détectée (par la caméra du robot Ergo Jr).

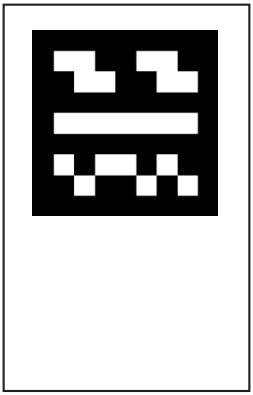
On peut par exemple créer ce script :



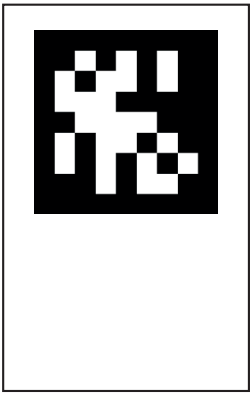
Code Caribou

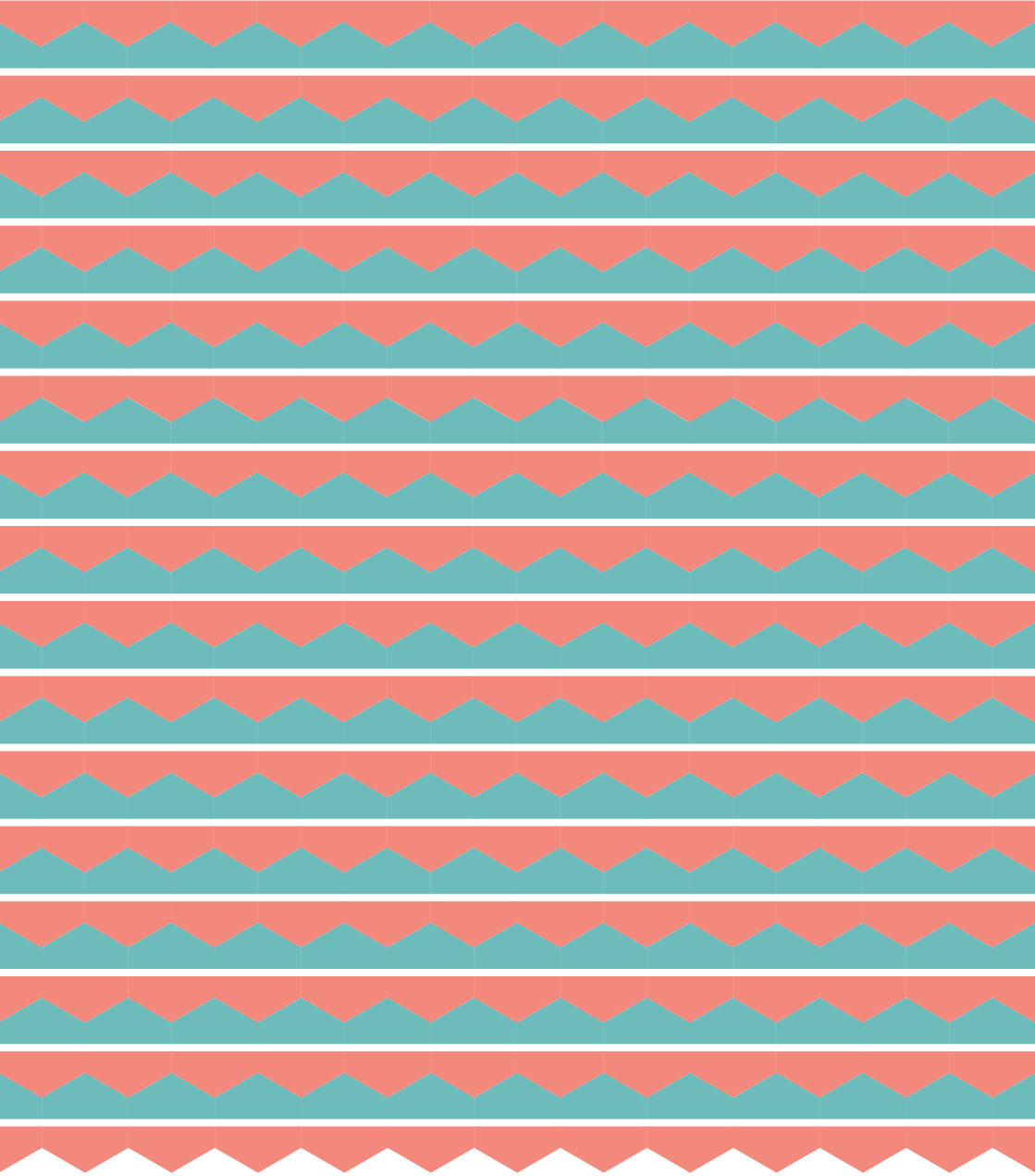


Code Tetris



Code Rabbit





Livret d'accompagnement du robot Poppy Ergo Jr

Poppy Project 2016

Version 1.0 (rev 0)