

À la découverte de Snap! avec Poppy Ergo Jr

un langage de programmation visuelle par bloc

1. Cliquez, cliquez, cliquez encore

Cliquez sur les blocs qui sont proposés sur la zone de script (zone du centre), observez, puis répondez aux questions.



Sélectionnez la ou les bonne(s) réponse(s).

Les blocs au bord	carré	rond	pointu	nous retournent une information.
Les blocs au bord	carré	rond	pointu	exécutent une action.
Les blocs au bord	carré	rond	pointu	réalisent un test.

Vrai ou faux:

Les blocs tests ne renvoient qu'un type d'information (un booléen) qui est soit vrai (*true* en anglais) soit faux (*false*).

2. Déplacez, cliquez, observez, recommencez

Sans modifier les chiffres inscrits, essayez de rendre les tests présentés "vrais" (*true*).



3. Assembler et contrôler

Le bloc tourner (*turn*) de 15 degrés (*degrees*)



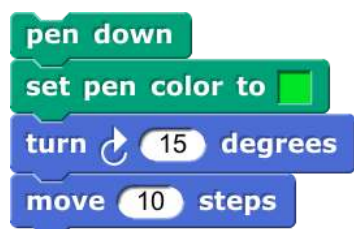
Et Le bloc avancer (*move*) de 10 pas (*steps*)



Vrai ou faux: Les deux blocs ci-dessus permettent de contrôler le robot Poppy Ergo Jr.

Les blocs au bord	rond	pointu	carré	peuvent s'attacher les uns à la suite des autres.
Les blocs au bord	rond	pointu	carré	peuvent s'insérer à l'intérieur d'autres blocs.

4. Répéter



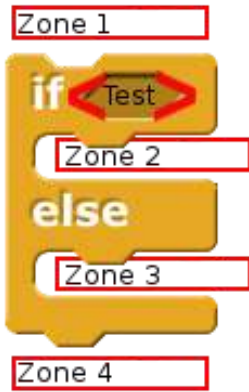
Combien de fois faut-il répéter la série d'instructions ci-contre pour dessiner un cercle entier? _____

Astuce:

Trouvez et utilisez le bloc répéter (*repeat*)
Indice: il se trouve dans la catégorie jaune "Control"



5. Conditionner



Si (**if**) le test effectué est vrai, alors exécuter des instructions, sinon (**else**) en exécuter d'autres. Voilà ce que vous permet le bloc jaune ci-contre.

Le bloc **dire (say) Bonjour! (Hello!) pendant 2 secondes (for 2 secs)** permet d'afficher un texte



placé dans la zone 1, il s'exécute uniquement si
 placé dans la zone 2, il s'exécute uniquement si
 placé dans la zone 3, il s'exécute uniquement si
 placé dans la zone 4, il s'exécute uniquement si

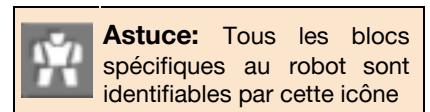
le test est vrai
le test est vrai
le test est vrai
le test est vrai

le test est faux
le test est faux
le test est faux
le test est faux

6. Et le robot dans tout ça

Pour programmer le robot Poppy Ergo Jr, certains blocs ont été ajoutés au bloc **Snap!** Par exemple:

- test connection** Permet de vérifier l'état de la connexion avec le robot.
- all motors** Permet d'obtenir la liste complète des moteurs du robot.



7. Agir

Que se passe-t-il lorsque vous cliquez sur les instructions ci-dessous ?



Vrai ou faux:

Le robot effectue une action: il s'allume en vert. Puis, après avoir attendu (wait) une seconde, il s'éteint.

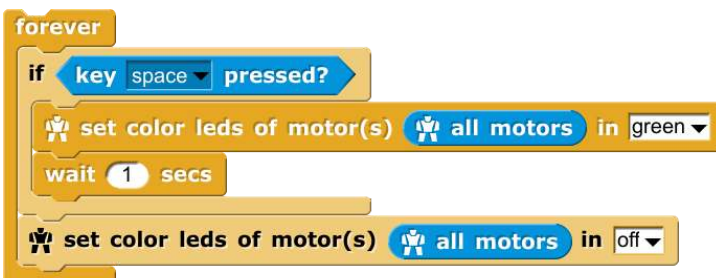
Dans cette série de 3 instructions, combien de blocs sont utilisés ?

3 4 5 6 7

Parmi ceux-ci, combien sont des blocs spécifiques au robot ?

3 4 5 6 7

8. Capter puis agir



Dans cette série d'instructions, combien de blocs sont utilisés ?

5 6 7 8 9 10

Parmi ceux-ci, combien sont des blocs spécifiques au robot ?

3 4 5 6 7 8

Que se passe-t-il lorsque vous cliquez sur les instructions ci-dessus?
 Appuyer (**press**) sur la touche (**key**) espace (**space**) du clavier pour en découvrir plus.

Astuce:

Le bloc pour-toujours (**forever**) répète une série d'actions indéfiniment.
 Cliquez une seconde fois sur ce bloc pour l'arrêter.



9. Des moteurs, mais pas que

Chacun des 6 moteurs du robots est en réalité un servo-moteur. Un servo-moteur est capable de maintenir une opposition à un effort statique ; sa position est vérifiée et corrigée en continu. Le servo-moteur intègre dans un même boîtier, la mécanique (moteur et engrenage), et l'électronique.

Il en résulte deux modes d'interactions: le mode *souple* (*compliant*) et le mode *rigide* (*stiff*)

➡ Clic sur ces blocs puis essaie de déplacer le robot avec tes mains.



En mode stiff	<i>on peut</i>	<i>on ne peut pas</i>	faire bouger le robot avec les mains.
En mode compliant	<i>on peut</i>	<i>on ne peut pas</i>	faire bouger le robot avec les mains

➡ À présent, que ce passe-t-il lorsque tu cliques sur cette suite d'instructions A:



En mode stiff	<i>on peut</i>	<i>on ne peut pas</i>	faire bouger le robot avec les blocs Snap!
En mode compliant	<i>on peut</i>	<i>on ne peut pas</i>	faire bouger le robot avec les blocs Snap!

Je peux donc déterminer (*set*), pour un servomoteur, une position spécifique à atteindre et à maintenir.

Grâce au bloc *get present_position of motor(s)*
Je peux également récupérer (*get*) sa position actuelle.



Astuce:

Modifie la valeur 0, 15 ou -15 située dans les blocs *set*, puis clic sur le bloc *get* pour connaître la position des moteurs. Remplace le bloc "all_motors" par m4 pour n'obtenir que la position du moteur m4.

En mode **compliant** les valeurs maximales pouvant être relevées sont:

min = ____ degrés *max= ____ degrés*

En mode **stiff** les valeurs maximales pouvant être atteintes sont:

min = ____ degrés *max= ____ degrés*

➡ Maintenant, crée une suite d'instruction B: modifie la suite d'instruction A comme ceci



Astuce:

Effectue un clic-droit sur la suite d'instruction A puis clic sur "dupliquer" (*duplicat*)

Combien de temps est nécessaire à la suite d'instruction **A** pour s'exécuter ? 1 2 3 4 5 6 *secondes*

Combien de temps est nécessaire à la suite d'instruction **B** pour s'exécuter ? 1 2 3 4 5 6 *secondes*

Quel est le nom de chacun des 6 moteurs ? _____

10. Agir ... Capter ... Réagir ... Interagir ... Un comportement pour un robot!

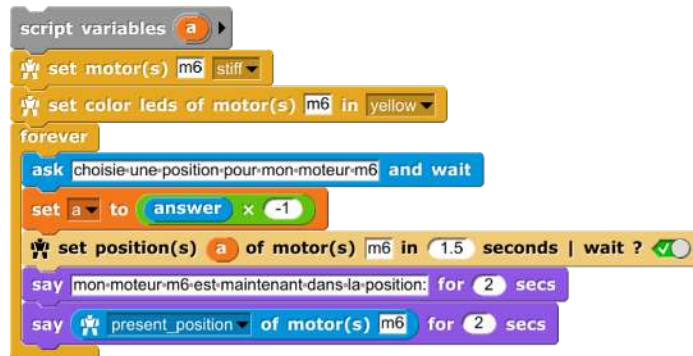
Un robot possède des capteurs, des actionneurs et un microcontrôleur contenant le programme qu'il exécute. De l'interaction entre le robot et son environnement émerge un comportement. Un comportement complexe peut émerger de règles extrêmement simple.

➡ L'esprit de contradiction ...

Reproduit la série d'instruction ci-contre:

Astuce:

a et **answer** sont des variables. Ce sont des sorte de boîte où l'on peut stocker des choses.



Que ce passe-t-il lorsque j'exécute cette série d'instructions:

- Le robot me demande quelles position affecter à son moteur m6 *vrai* ou *faux*
- Ta réponse est stockée dans la variable nommée "answer" *vrai* ou *faux*
- On calcule l'inverse de ce qui se trouve dans la variable "answer" *vrai* ou *faux*
- La variable "a" contient la valeur opposée à ta réponse initiale *vrai* ou *faux*
- Le robot t'indique qu'il a atteint la position que tu lui a demandé *vrai* ou *faux*

Quelle(s) valeur(s) te permettra de piéger le robot pour qu'il atteigne la position que tu lui as demandée? _____



Défi robotique: Aussi stable qu'une poule!

Avez-vous déjà manipulé une poule ??? Non! Alors allez voir:

➡ <https://frama.link/ergo-poule>

Principe:

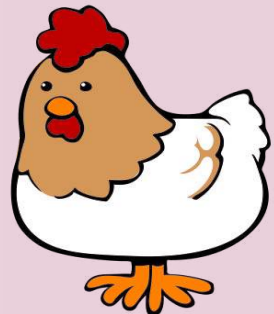
la partie inférieur de son corps bouge, sa partie supérieur bouge également et compense les mouvement de la partie inférieur stabilisant sa tête.

Objectif:

Créer un programme permettant de garder l'extrémité supérieur du robot toujours dans la même direction malgré vos manipulations sur sa base.

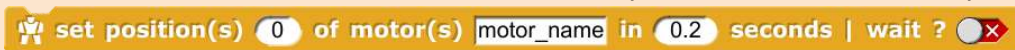
Construisez votre programme:

Les moteurs m1 m2 m3 sont en mode compliant et de couleur verte ; m4 m5 m6 en mode stiff et de couleurs rouge. Pour toujours, récupérer les valeurs des 3 moteurs inférieurs, calculer leurs opposé, envoyer ces nouvelles valeurs aux 3 moteurs supérieurs.



Astuce:

La connexion entre votre ordinateur et le robot n'est pas instantanée! En moyenne, il faut 0.2 seconde entre le moment où vous interrogez le robot et le moment où vous recevez sa réponse. Pour donner l'illusion d'un mouvement fluide, préférez la valeurs de 0.2 plutôt que 0.



10. Agir ... Capter ... Réagir ... Interagir ... Un comportement pour un robot!

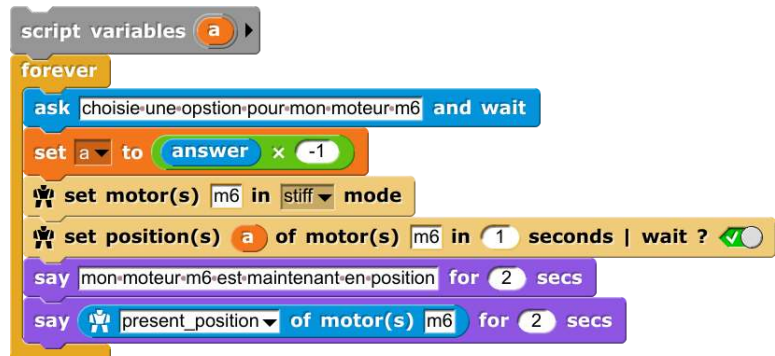
Un robot possède des capteurs, des actionneurs et un microcontrôleur contenant le programme qu'il exécute. De l'interaction entre le robot et son environnement émerge un comportement. Un comportement complexe peut émerger de règles extrêmement simple.

➡ L'esprit de contradiction ...

Reproduit la série d'instruction ci-contre:

Astuce:

a et **answer** sont des variables. Ce sont des sorte de boîte où l'on peut stocker des choses.



Que ce passe-t-il lorsque j'exécute cette série d'instructions:

- Le robot me demande quelles position affecter à son moteur m6 vrai ou faux
- Ta réponse est stockée dans la variable nommée "answer" vrai ou faux
- On calcule l'inverse de ce qui se trouve dans la variable "answer" vrai ou faux
- La variable "a" contient la valeur opposée à ta réponse initiale vrai ou faux
- Le robot t'indique qu'il a atteint la position que tu lui a demandé vrai ou faux

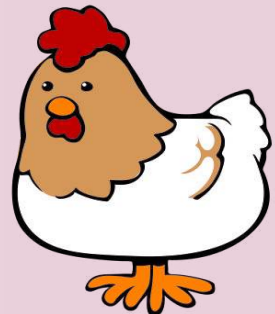
Quelle(s) valeur(s) te permettra de piéger le robot pour qu'il atteigne la position que tu lui as demandée? _____



Défi robotique: Aussi stable qu'une poule!

Avez-vous déjà manipulé une poule ??? Non! Alors allez voir:

➡ <https://frama.link/ergo-poule>



Tu peux essayer de reproduire ce comportement avec le robot !

Clic sur le bloc:



Dans quel mode sont les 3 moteurs inférieurs (en vert) ?

compliant *stiff*

Dans quel mode sont les 3 moteurs supérieurs (en rouge) ?

compliant *stiff*

Avec tes mains, que ce passe-t-il

- lorsque tu bouge m1 (et uniquement m1) ? _____
- lorsque tu bouge m2 (et uniquement m2) ? _____
- lorsque tu bouge m3 (et uniquement m3) ? _____

Construit une suite d'instruction produisant le même comportement que celui que tu viens d'observer.

Astuce:

La connexion entre votre ordinateur et le robot n'est pas instantanée! En moyenne, il faut 0.2 seconde entre le moment où vous interrogez le robot et le moment où vous recevez sa réponse. Pour donner l'illusion d'un mouvement fluide, préférez la valeurs de 0.2 plutôt que 0.

