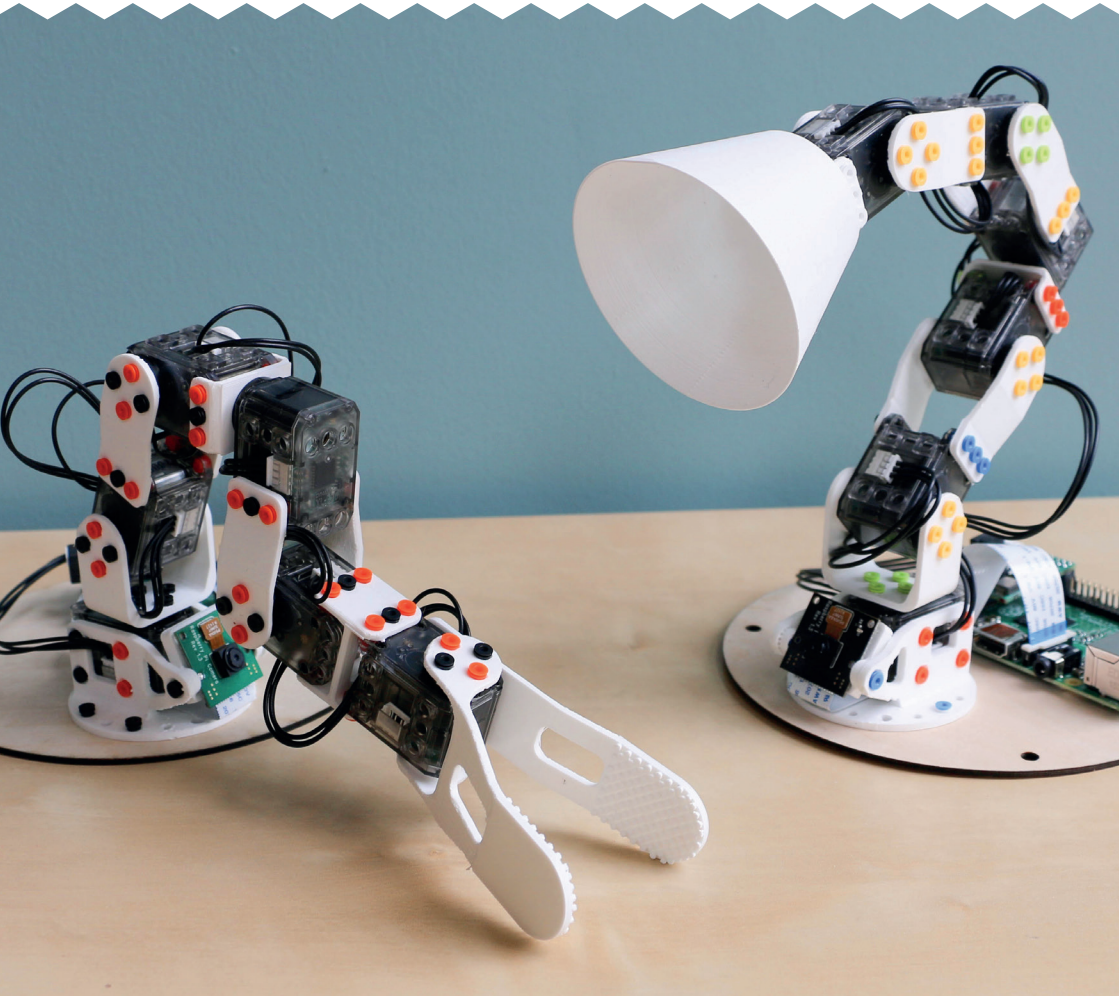
Learning to program Ergo Poppy Jr Snap!

This booklet offers activities and small challenges to achieve to become familiar with the Poppy Ergo Jr robot programming language Snap !.



for Education
poppy

Feel free to join us on our forum (forum.poppy-project.org, Education category)
to share your productions and ask any questions you want!

Learning to program Ergo Jr Snap!

accompanying booklet Poppy Ergo Jr robot

Version 1.0 (rev 0)

The Poppy Ergo Jr robot was created in the project Poppy Education Research Team FLOWERS (INRIA, ENSTA Paris Tech), supported by the Aquitaine Region and the European Funds ERDF.

Contributors to the book:

• Design and production: FLOWERS Team (INRIA, ENSTA Paris Tech): Stephanie Noirpoudre Didier Roy, Marie Demangeat Thibault Desprez, Theo Segonds Pierre Rouanet, Damien Caselli, Nicolas Rabault Matthieu Lapeyre, Pierre-Yves Oudeyer. Websites: fl owers.inria.fr and www.poppy-project.org

• Graphic design and layout: Antonin + Margaux

Thanks

At the Rector of the Academy of Bordeaux and teachers working group Education Poppy Project who helped design and test activities and robots in the classroom (Benrahho Said Youcef Bouchemoua Christophe Casseau Olivier Eloi, Armelle Grenouilleau, Nicolas Griff e Gilles Lassus, Georges Layris Sebastien Prouff Joel Rivet Thierry Salem, Sylvain Soulard, Luc Vincent). Generation robots for supporting the Poppy project.

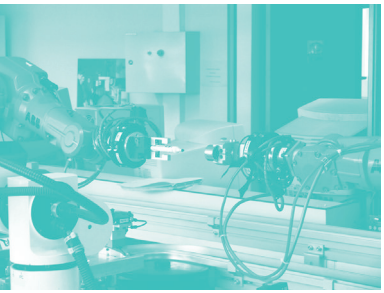
Translated to english by Jeff Brodsky

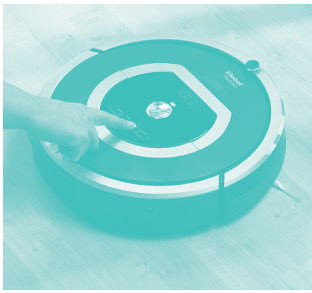
Summary

Introduction

What is a robot?

A robot is a machine with engines that allow it to move and / or act on its environment sensors that allow it to charge his own state and its environment, and an electronic or computer system that controls what the qu'effectue robot according to what it perceives.





You can see the robots in many areas, for example:

- **Industrial robots:** relieve repetitive or dangerous tasks of workers (eg nuclear industry) or assist in procedures that require a level of precision or speed inaccessible to human.

- **Robots of intervention:** used to perform tasks in environments where humans can not go or dangerous for him (eg space).

- **domestic robots:** perform independent tasks in the home or have a direct relationship with the occupants (eg, robotic vacuum cleaners, robots tele- monitoring ...).

Robots are also used to represent certain aspects of animal and human human behavior, such as the mechanics of walking or learning the language. This may help scientists better understand these behaviors by testing with the robots if they imagine the mechanisms to explain these behaviors can work. With robots, eg they can test how the information collected by the senses (sight, hearing) are combined to produce actuators allowing the body to move.

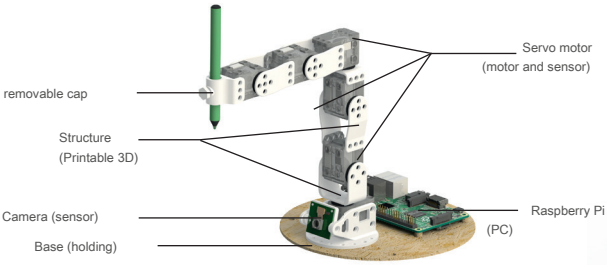
source:
Oudeyer, PY Where are the robots? *pyoudeyer.com/LettreMURS32.pdf*

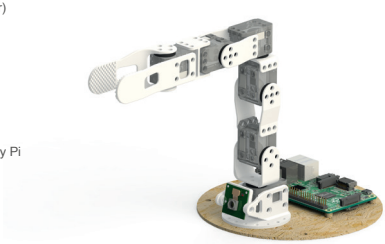
The ancestors of the robots are robots, which are eff machinery ectuant a specific task and always in the same way. **Unlike the controller,** the same robot behaves so diff erent according to what is happening in its environment.

Robot Features Poppy Ergo Jr

Poppy Ergo Jr is a small open-source of the poppy family robot. His pieces are printed with a 3D printer.

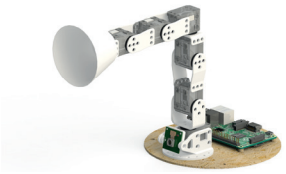
The Poppy robotic platform that comes Poppy Ergo Jr robot is open- source software and hardware, that is to say, its software and its plans **available publicly, freely used and modified reliable.** With this approach, legal and philosophical, collaborations settled around projects and allowing for improvements and new developments. It is a powerful engine of development of new technologies.





It features :

- **sensors** to make information in its environment: its camera and its **servomotors** detect what is going on around him or her (its position, temperature etc.)



a servomotor is the combination of an engine and an electronic card performing precise control (servo) motor in a position, speed or given force.

• **On actuators** to produce actions: its **servomotors** allow moving and its LEDs emit light

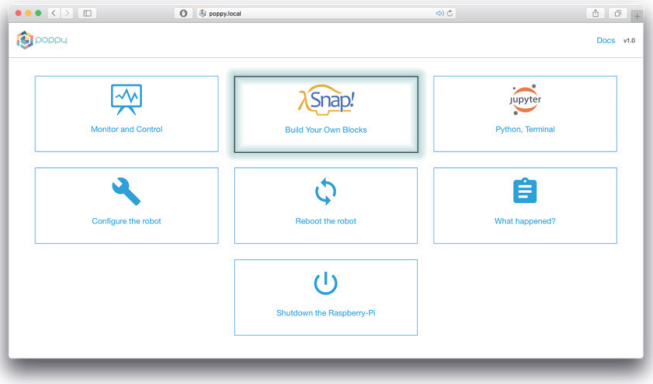
• **On the computer:** connected to the sensors and actuators, it allows **control** the robot behavior by running a computer program that contains instructions

In a robot, there is a computer, sometimes small. This is the same type of computer is used in everyday life (those with a keyboard and a mouse). It controls the robot, give instructions and record the data. For Ergo Jr, this computer called Raspberry Pi.

Begin by Snap!

To do before you start: build, connect your Poppy Ergo Jr robot, and connect on its home page.

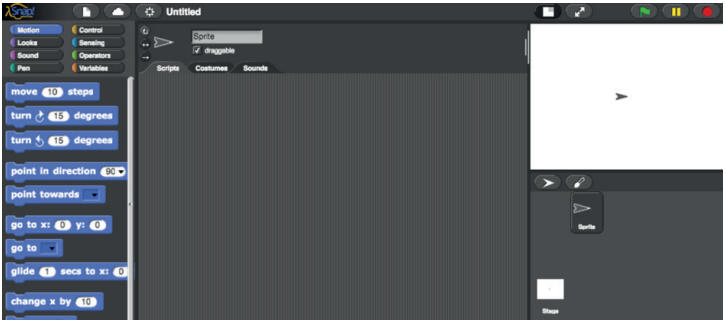
AT Click on the box Snap! the home page:



The interface Snap! is divided into three parts:

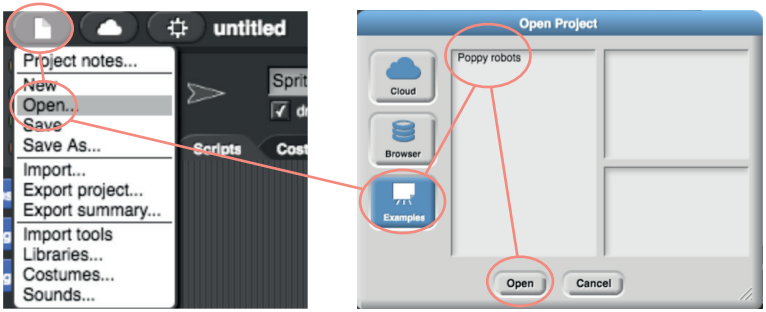
- **To the left :** the list of blocks (instructions) available, stored in diff erent categories.
- **In the center :** Zone scripts, which are available and assembles the blocks to make programs.
- **To the right :** a top area ffi controllable drying by scripts, bottom sprites, programmable objects may change in the a ffi drying area.

To program Poppy Jr Ergo, you mainly use areas left and center.



B Open the project *Poppy robots* to access the specifi c blocks at Poppy robot:

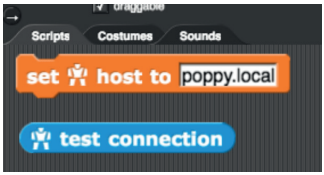
To do this click on File> Open> Templates> Poppy robots> Open



(Continued on next page)

Tip: For information on a block Snap! *right click* on the block and select *h elp*.

You should see two blocks scripts in the area:



C Should see your you are connected (e) the robot:

1. Write the name of the robot or the IP address in the block 

:

(Here the name of the robot *poppy)*

2. Click the block 

to inspect to see that you are connected the robot:

You should see your Connections (see the documentation of the robot, if necessary)

D Save your program:

1. You can save it on the hard drive of your computer by clicking on:
Files> Export> right click> Save As

2. Or save it in the cloud, which needs to have a user account: If an error message appears:

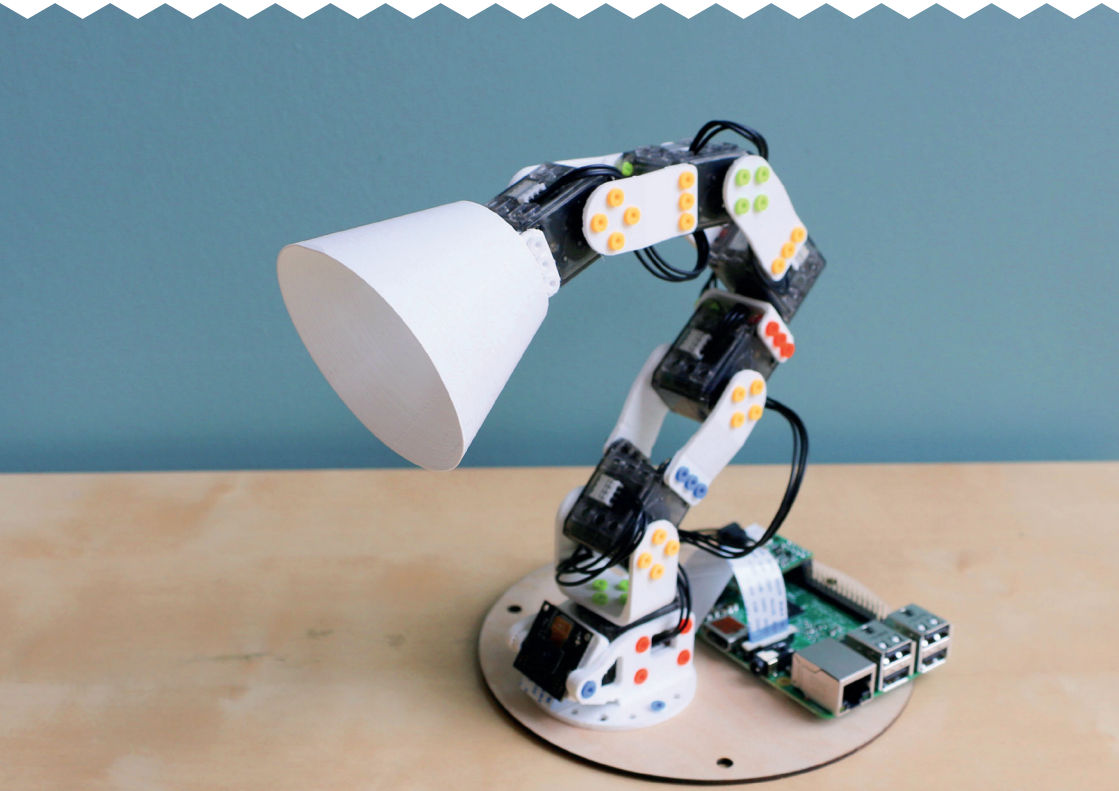
The cloud storage signifi e that all information that you visit is stored on servers (computers) at various locations around the world.

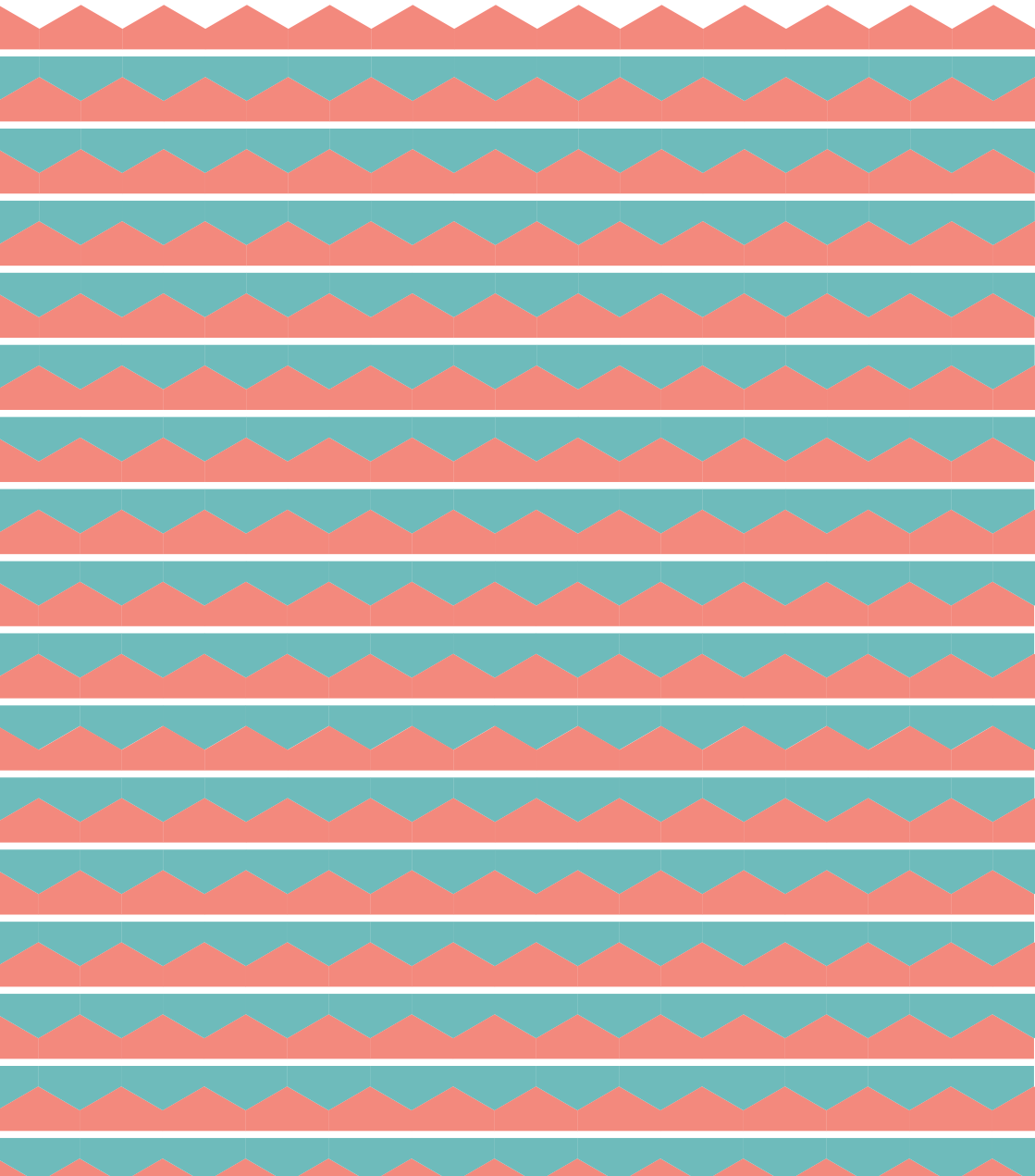
- Click the menu *cloud (* cloud) in the toolbar:

- Select *sign up* in the menu, and follow the instructions.
- Should see your email to get your initial password.

Before checking Ergo Jr. should see your every time the correct position of the robot.

You are ready to start work!





Part 1 :

Check Poppy Ergo Jr

You will learn to move the Poppy Ergo Jr robot using the programming language Snap !. At the end of the session you will apply your knowledge with the challenge **Ergo Jr plays coconut shy.**

A **programming language** You can write a computer program, which is a sequence of instructions to execute. This will give a robot behaviors. Snap! is a programming language with which assembles *instruction blocks.*

An assembly of blocks is called a *script.*

Your first program

Finding blocks in Snap !, search:

• By **color / category (** each category a color):

• By **keywords (>** Right click on the left side> *fi nd blocks)*

and enter:

- A word in the desired block (example: *When)*

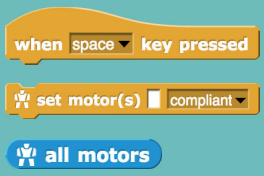
- The key word *robot* afi n to select only blocks specifi c robot Ergo Jr

It's your turn !

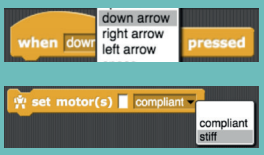
AT Create two scripts below afi n to be able to Ergo Jr in specifi c positions:

For that :

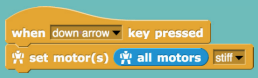
1. Select and drop the three blocks that you need on the central workspace:



2. Click on the lists and choose the right values:

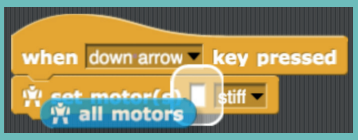


3. Assemble the blocks together:



4. Do the same for the second script.

To max out the blocks: Select the block and drag it to the desired place with the mouse. The white border indicates that the blocks will follow suit.



You can copy / paste blocks and scripts: *Right click* then *duplicate*

B Enable both scripts (a white border appears around a script on):

1. Press on ⇩ on your keyboard and then manipulate the robot. What do we NOTE-?

2. Press on ⇧ on your keyboard and then manipulate the robot. What do we NOTE-?

C What mode *compliant* the robot? And mode *sti ff?*

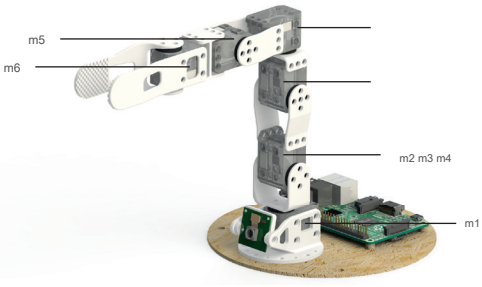
D Enable these scripts and handle Ergo Jr. to put it in diff erent positions.

Ideas : Give him curiously, shy, happy. As you imagine!

Ergo Jr move using its engines

Is used the next block to move Ergo Jr, engine output:

Here the robot diagram with the name of each of the motors:



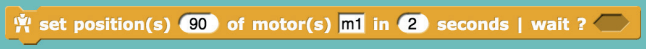
> **i** It is **necessary** activate the motors (**stiff)** before the robot to be able to **move** with Snap!

It's your turn !

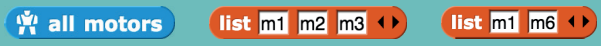
A **Make sure all motors are properly activated (fashion *sti ff)*** and put all motors in the basic position (which corresponds to the position where each motor is at 0 degrees: aligned with the notch) by selecting the next block to execute:

> **i** Here the block is used *set position (s):* it accepts in degrees position values, it is recommended that a range of [-90; 90], followed by / name (s) of / engine (s), then the time in seconds it will take to reach that position. Regarding the value *wait?* we will use later.

B Start the engine *m1* in the 90 degree position in 2 seconds.



C Look for the following blocks and run them against each in turn:



1. What values these blocks they refer?

i

The blocks have diff erent forms, each form is a category specifi.

• oval shaped blocks (as 

) are called *report :* when it is executed, they return a value.

• At the top of the script can be a block *Hat (* hat), which indicates when the script should be executed. The block names *Hat* usually begin with the word *When (* example:

); script does not necessarily block *Hat* but this block, the script will be executed only if the user clicks on the script itself.

• the blocks *command (* as correspond to an action. 

)

2. Ez modified block *set position* to start the engine *m1* and motor *m6* in position -30 degrees in 2 seconds.

D By helping you blocks that we have just discovered, build two programs corresponding to the instructions below:

1. When ⇨ on your keyboard is pressed then put all the engines in position 0 degrees in 3 seconds.

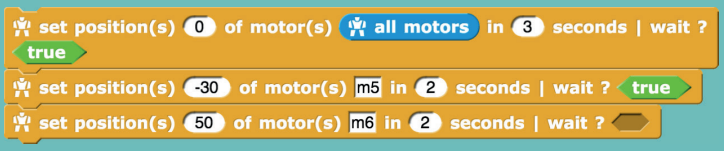
2. When ⇦ on your keyboard is pressed then put the engines m1 and m4 in position 60 degrees in 2 seconds.

Create movements

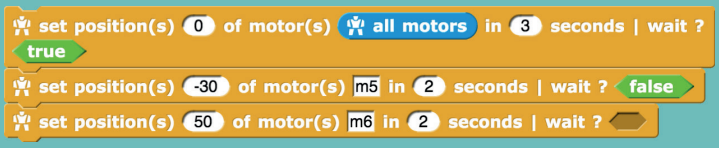
Now we will use the engines to create movements.

It's your turn !

A Run the following script and see what happens:



B Replace the second block `true` by block `false`



1. When there are two blocks (or more) nested in what order eff ectuent actions?

2. What will happen when *wait* is equal to *true?* What will happen when *wait* is equal to *false?*

i The lines of code run in an almost instantaneous; and although at times the required position in the previous line was not reached. The part *wait* makes it possible to wait until the engine has reached the desired position before executing the next command.

C With the blocks that you now know, play a movement to Ergo Jr signifi ant "hello" when you press the button *b.*

Advice:

- Start with a simple movement and enrich it to As.

- Select the engines you want to use to create the movement.

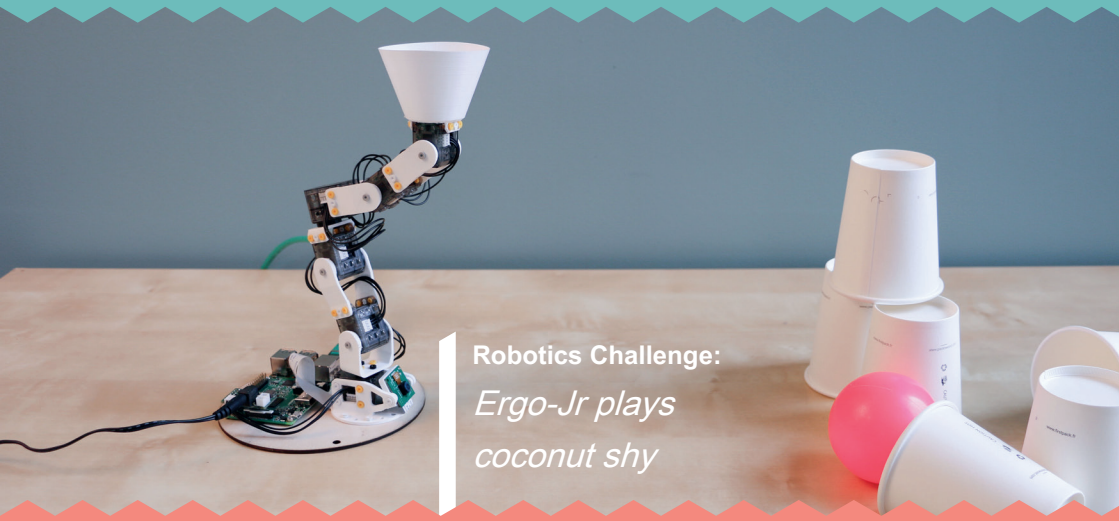
- **Play the movement chose the robot (mode *compliant)***
and observe the actions of each engine.

- You can help block 

to know the position of a target motor, and so note the value for reuse later.

- Program the movement motor engine and test each time the result of your program.

Feel free to create other movements!



Robotics Challenge:
Ergo-Jr plays coconut shy

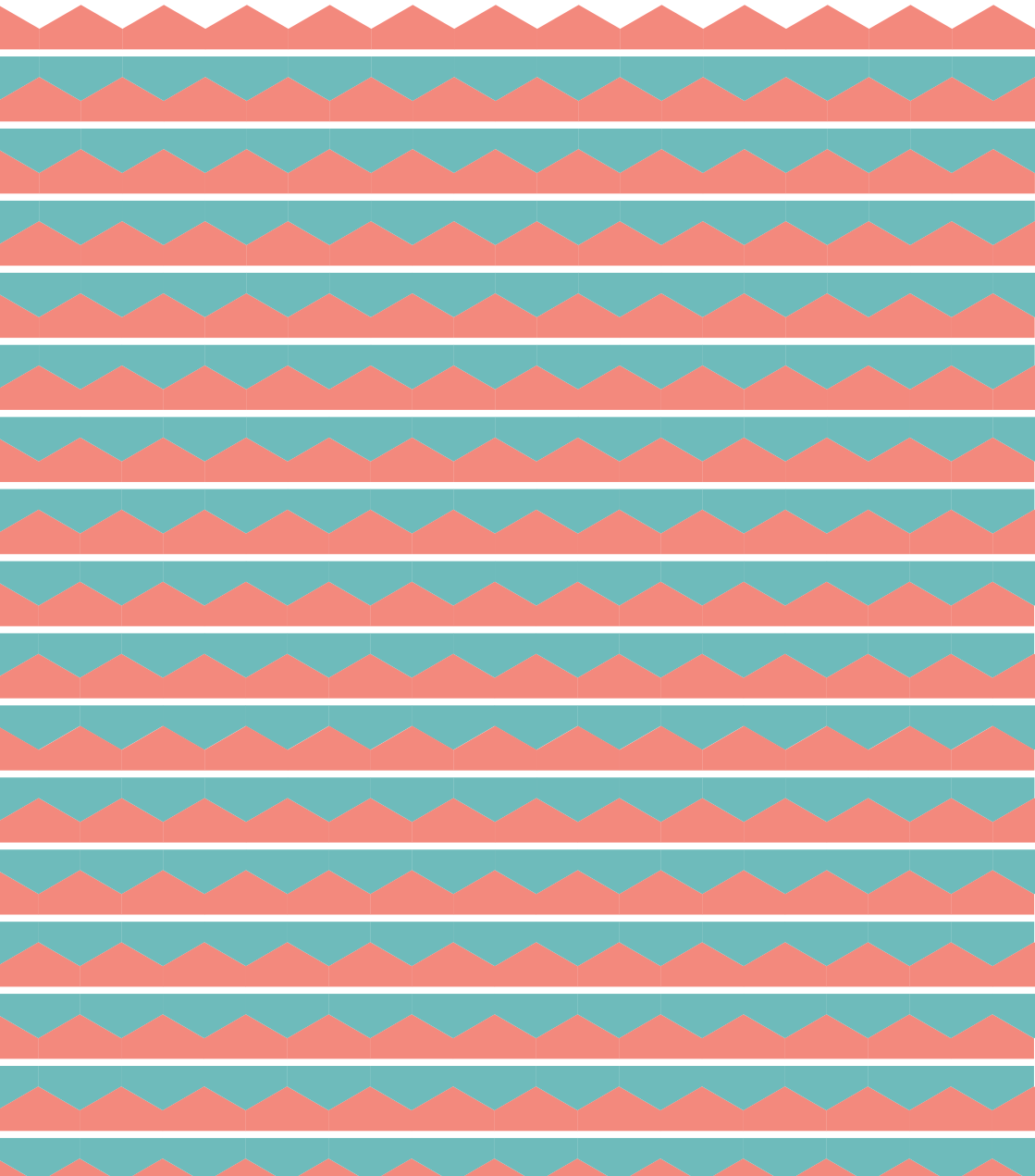
Equipment:

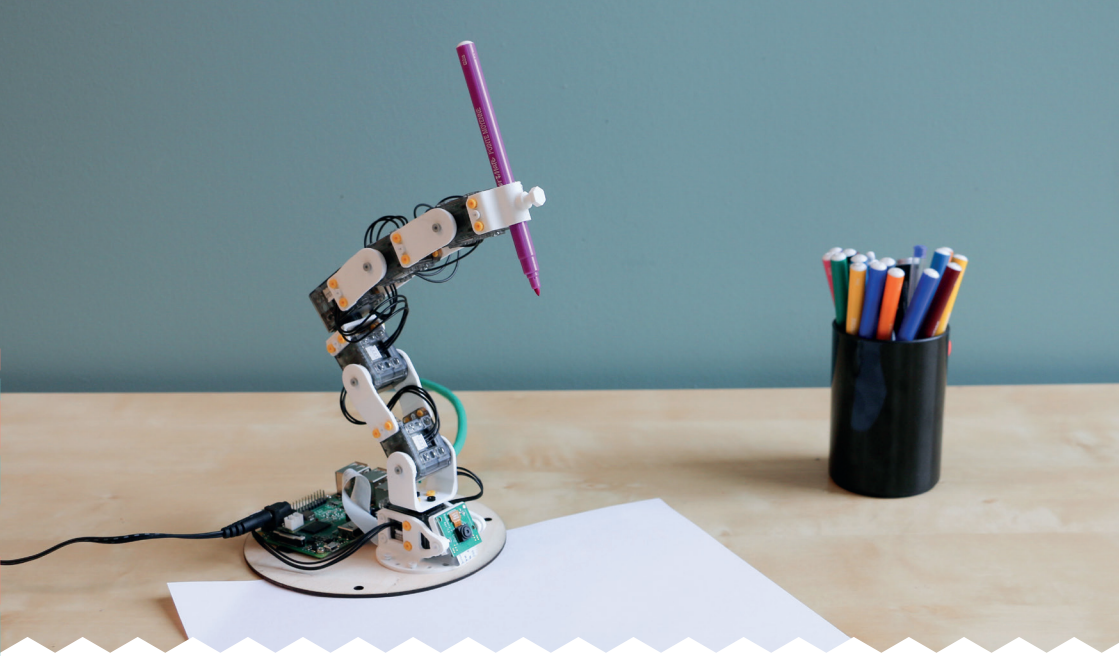
- Poppy Ergo Jr with lampshade
- A light ball
- Cups (cardboard or plastic)

Goal:

Check the position and speed of the robot's motors to throw the ball and drop the coconut shy.

There are many possible ways to throw the ball. How many can you find?





Part 2 :

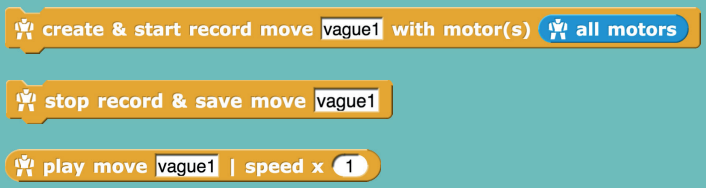
Programming by demonstration

Poppy Ergo Jr robot is capable of measuring in real time the position of its engines. Thus, when the moves manually, it can

Register movements made to reproduce subsequently.

It's your turn !

AT With the three blocks below record a wave motion and play it then:



For that :

1. Create and execute the block *create & start* to begin recording

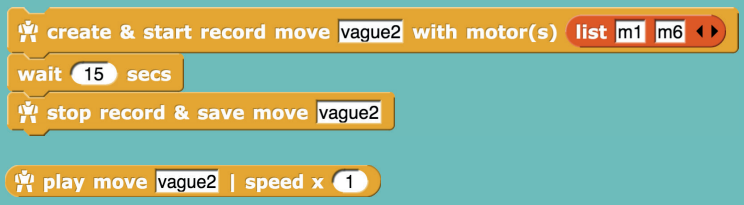
2. Handle the robot to create a movement to reproduce

3. Stop recording with the block *stop record*

4. Play the movement that you have created with the block *play move Feel free to repeat the movement until it suits you.*

> *i* The movements are saved in a file that is in the robot computer must Ergo Jr. **give a unique name to each movement** not eff acer fi le precede re-registering a movement from above.

B Watch the program below without creating and try to guess what it does:



C Create and test the script to inspect to.

To you ! Record the movement of your choice.

For further...

Many playback options are available: Experience them! What is happening in the following cases:

AT Ez modified in block 

the value of *speed* (with a *decimal or not* from 0 to 4)?

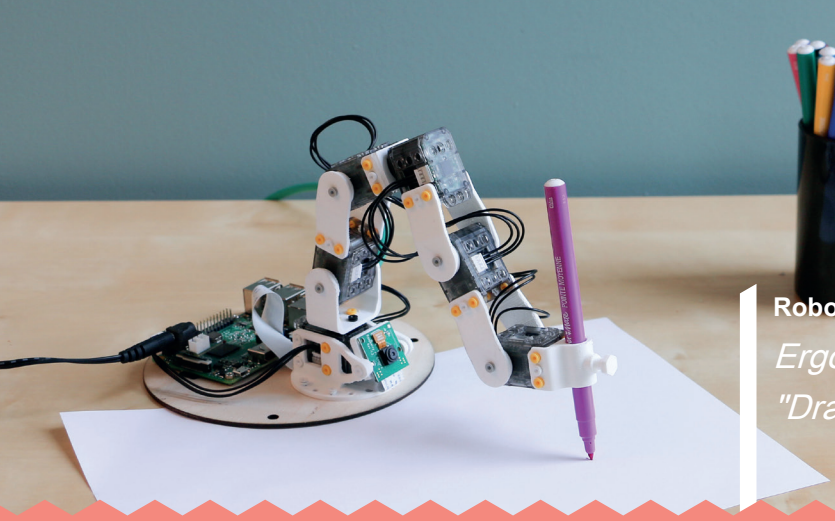
B Use the block 

C Use the block 

D Play three diff erent movements with the block

E Play a movement that has been registered with the engine m1 (*mouvement_m1)* and a second movement that was recorded with the m5 and m6 engines (*MoveMent_ m5_m6)* with block

In robotics, the position measurements are never perfect. This is also the case on the Ergo Jr robot, it is possible that the movement replayed does not exactly match what was shown.



Robotics Challenge:

Ergo-Jr plays the "Draw is Winning!"

Equipment:

- Poppy Ergo Jr with pencil holder
- Felt
- A list of words. Examples: sun, daisy, truck, snail, cat, beach, etc. A complete list is appended to the end of the booklet.

Goal:

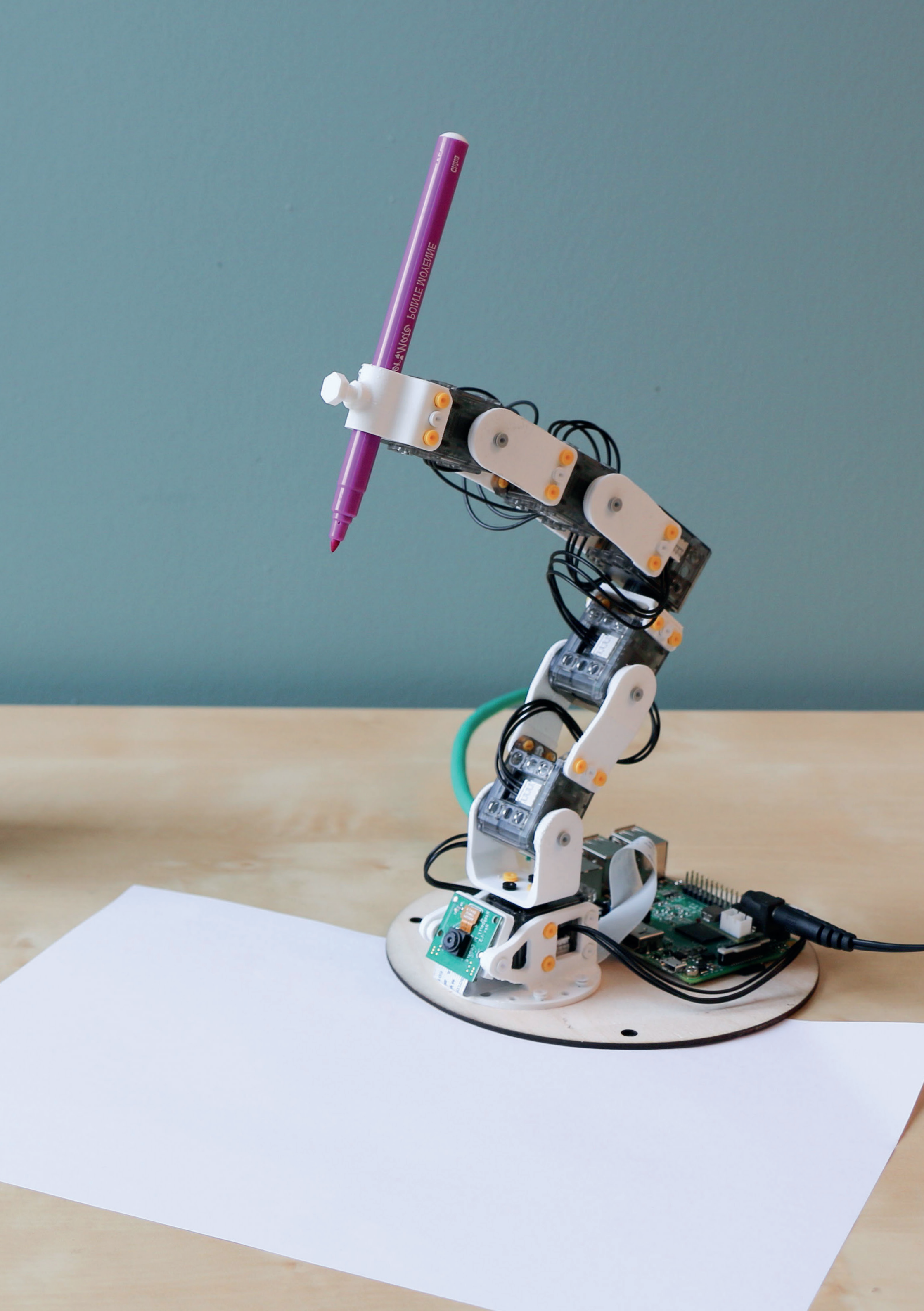
Preparation phase :

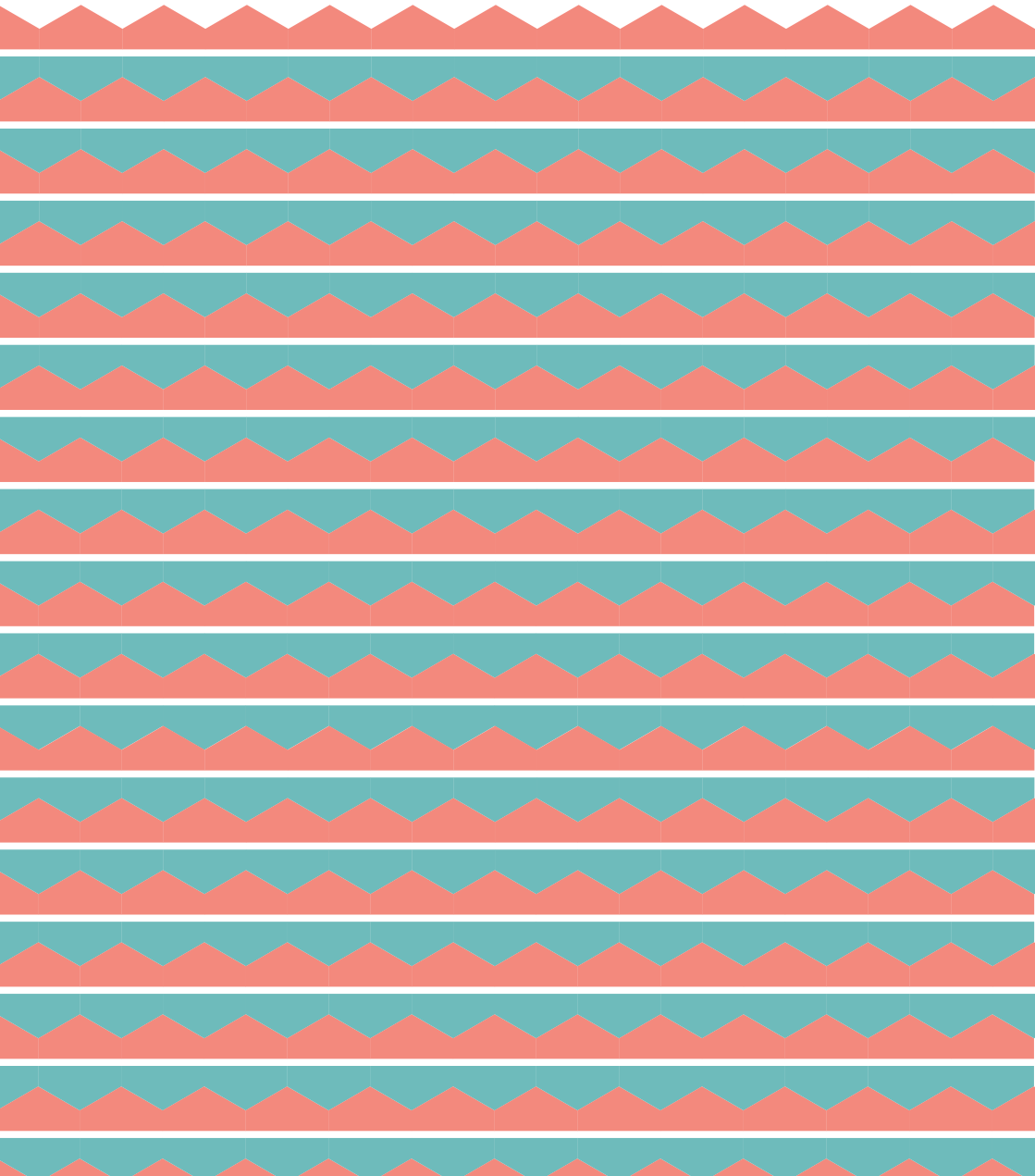
- Each team draws lots of bits of paper from the list of words
- In a limited time (eg 5 min), save the maximum of drawings by programming by demonstration

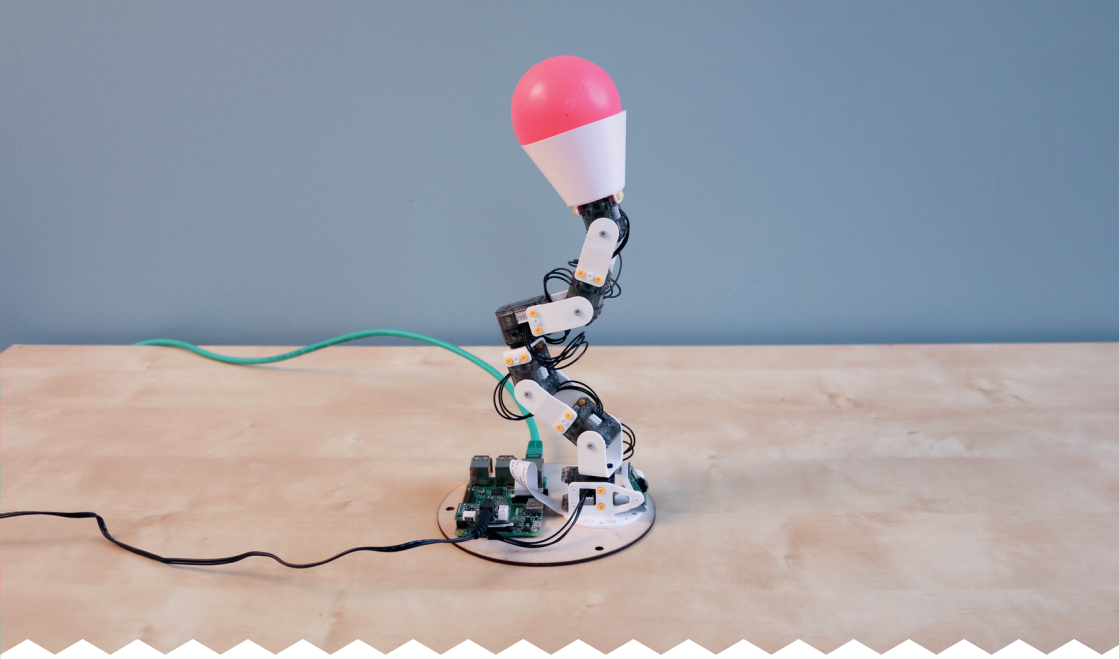
Phase of play:

- Each team is guessing other words illustrated in a limited time (eg 40 s)

- If the word has been guessed:
  - The person who guessed scores two points
  - Each person in the team scores a point
- If the word is not guessed, no mark point.







Part 3:

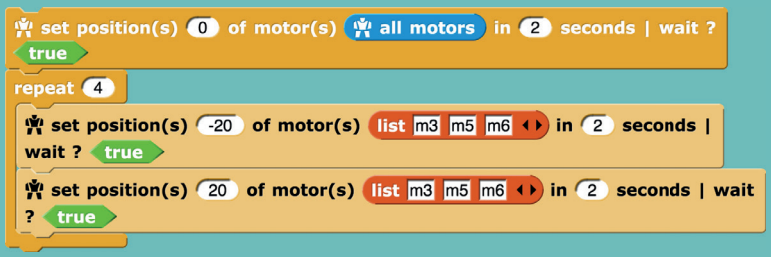
Use repetition

You will discover on this occasion an essential concept **in computer science and robotics: buckles ! You're** going to dance with the Ergo Jr block by repeating several times the same action.

We repeat !

It's your turn !

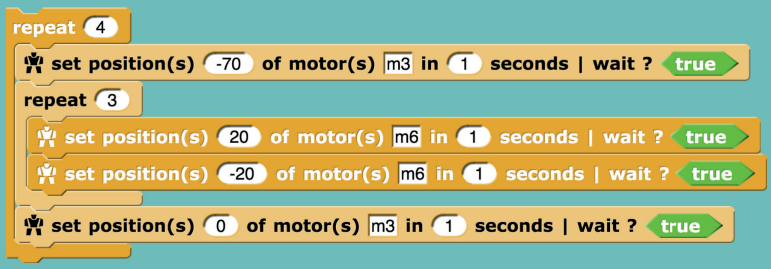
A Create and run the following script and see what happens:



1. What will happen if we change the value of the block 4 *repeat* by another?

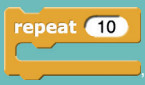
2. In your opinion, what are the interests of diff erent block *repeat?*

B Observe the following script without creating and say what it does:



1. How many times the engine *m3* Will he move?

2. How often the engine *m6* Will he move?

C Using the block 

, create a movement for the robot you do a little hello to congratulate you.

With loops, here Repeat block, scripts are shorter, often bring more clarity and also help to make more elaborate things.

Action!

It's your turn !

A Look for the block  and run it several time to try it.

1. What value does it refer?

2. Ez modified block so that it returns a value between [-80; 80]

B Now, use what you just learned to dance Ergo Jr randomly.

For that :

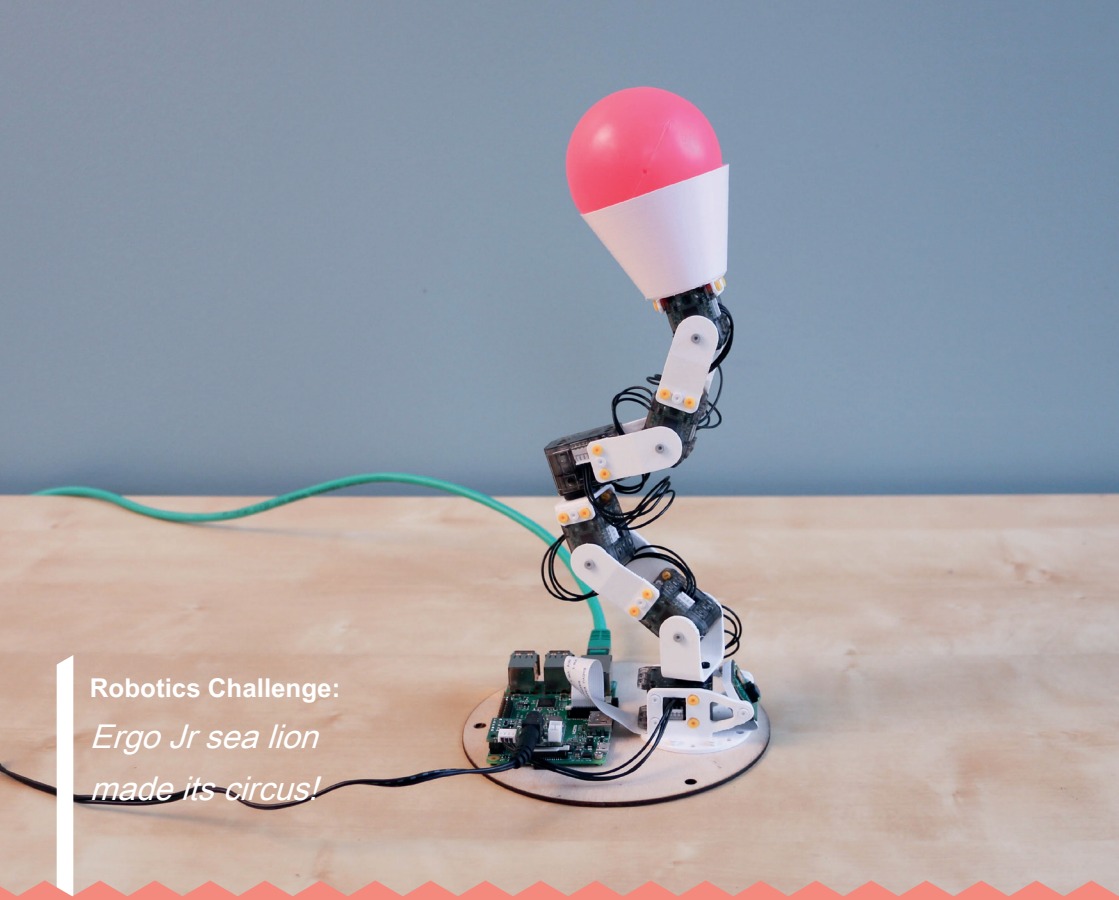
1. Start the engine *m1* to a random position between -80 and 80.

2. Do the same with the engine *m3* by choosing a range of positions you're comfortable (if necessary, help you block ).

3. Using one (or more) block *repeat,* make move **each motor one after the other (** choose well intervals).

For further...

Experience the blocks below and create scripts using:



Robotics Challenge:

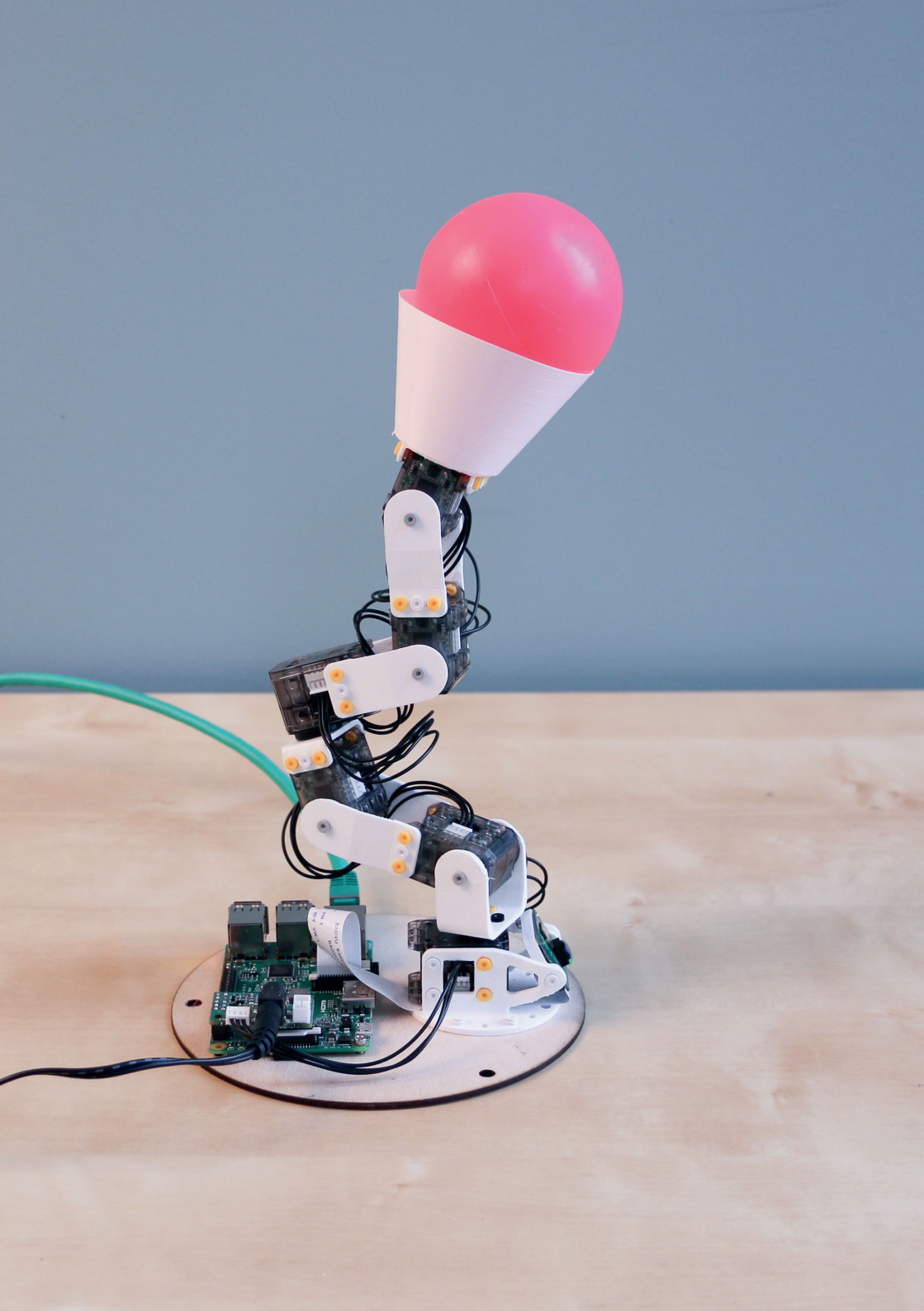
Ergo Jr sea lion made its circus!

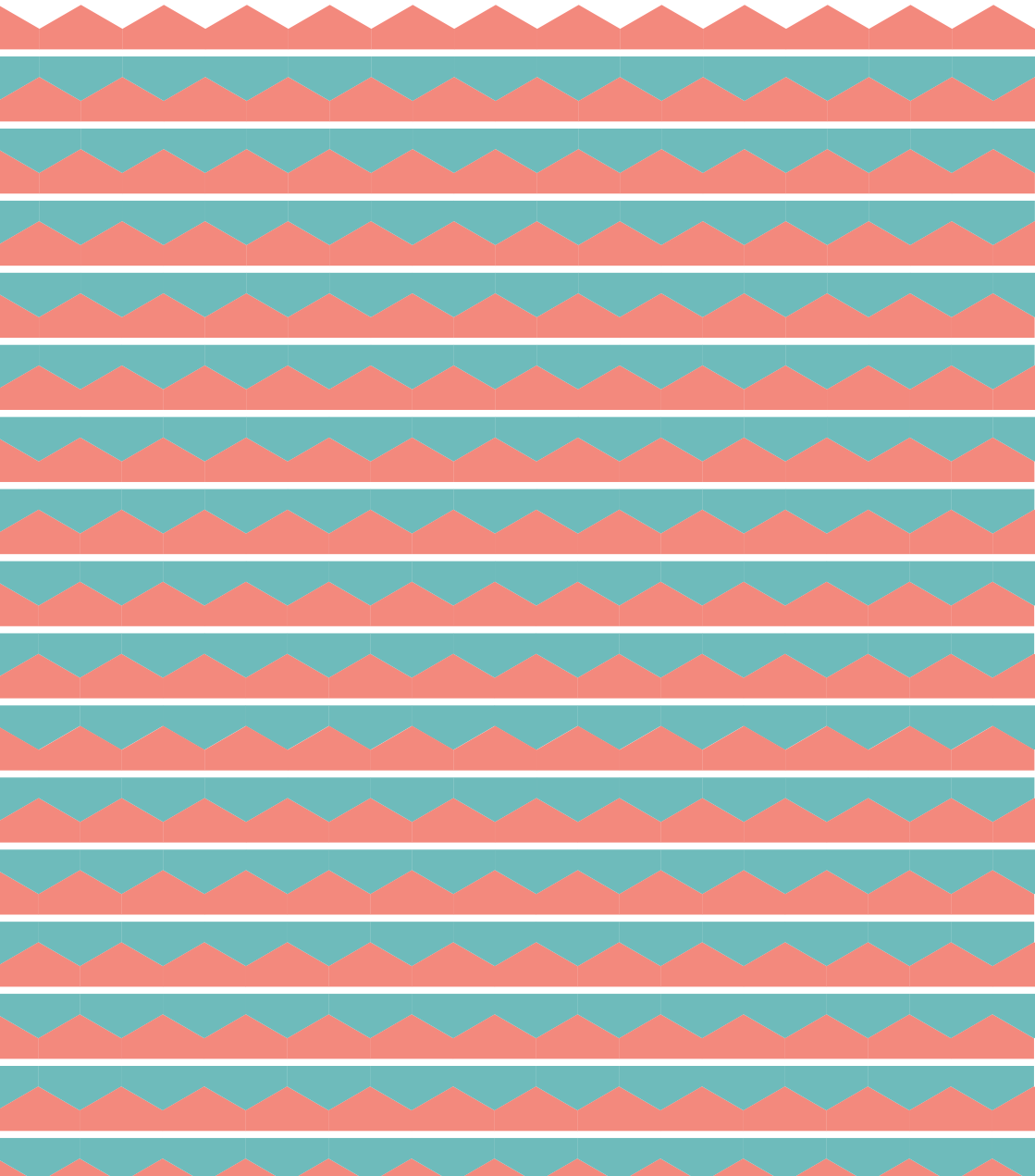
Equipment:

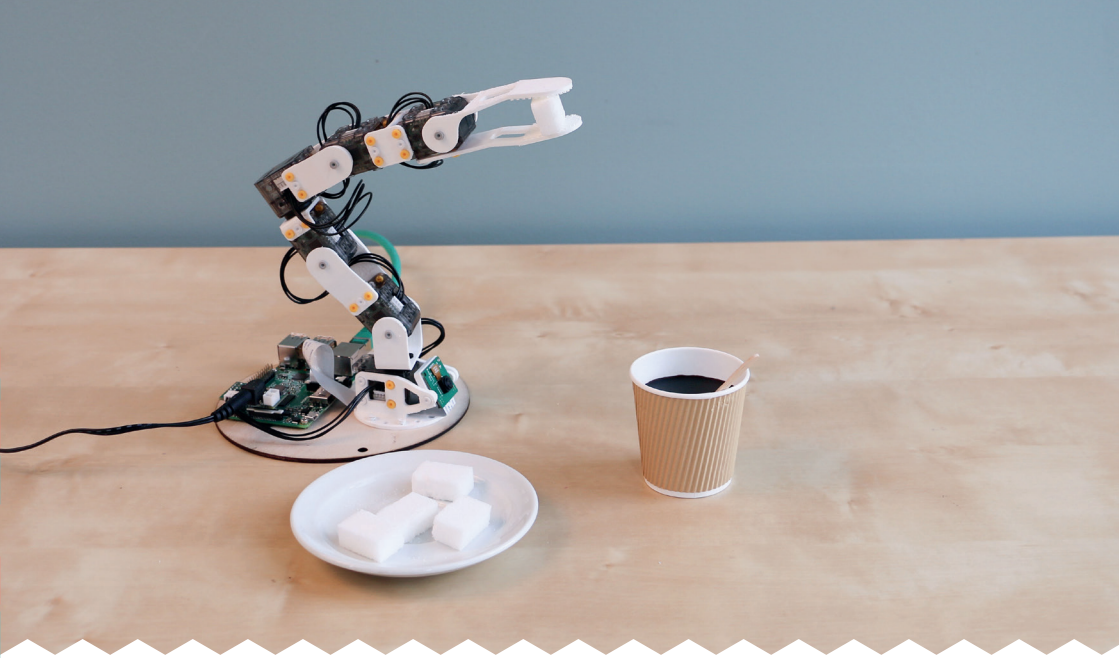
- Poppy Ergo Jr with the lampshade
- A light ball

Goal:

- Put the ball in the shade and make dance Poppy Ergo Jr randomly with repetition of movements. Use all engines, careful not to drop the ball!







Part 4:

Create your own Snap block!

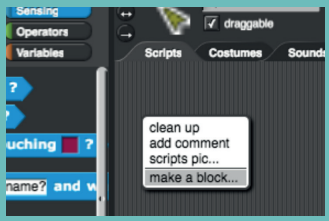
You will learn here to create your own blocks to store, reuse and modify behaviors that you have created.

Create Block

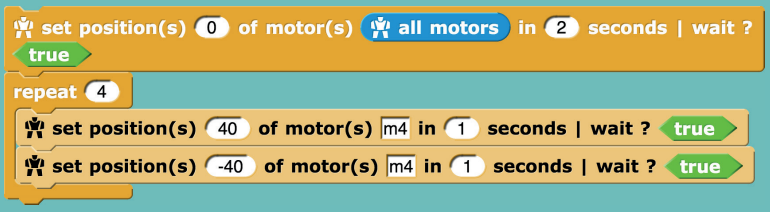
It's your turn !

In Snap! each block needs a color, title, category (form), and a script that defi nes its behavior.

A Right-click an empty area of the script box and select *make a block ...:*



B Create block kidinon will launch the script below:



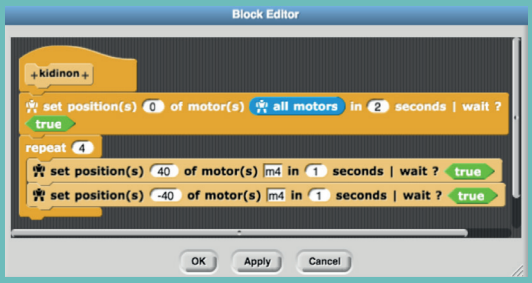
For that :

1. Choose the category *Control* yellow for storing your new block.

2. Give the block a name that describes the action of the script.

3. Select category *Command* (Because we want a block that is)

4. Building in the area *block editor* the script of your new block and confirm. (*page below cons)*



You can search your block (by category or by name) and use it in the same way as the others.

Add entry (*input)*

We use *an entrance* asking the user accurate information or to indicate an action; Here are blocks with inputs:

It's your turn !

A Add an entry to your block to allow the user change or alter the number of repetition of the movement *kidinon:*

1. Right-click the block and select *edit:*

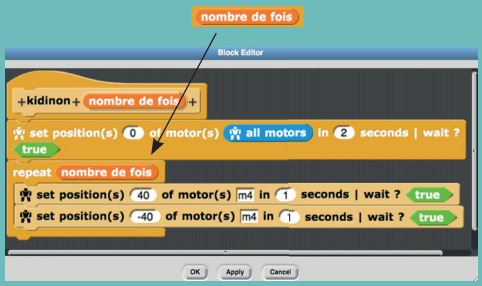
2. Click the + to the right:

3. Let the button *input name* selected and write *number of times :*

(Continued on next page)

4. You can now drop the variable *number of times* on the block input

repeat:



A variable is a memory area designated by a name that may contain values of one type (see section 7).

5. Validate and test your block 

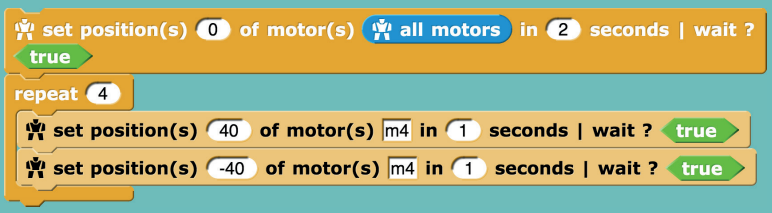
B Add the word again: 

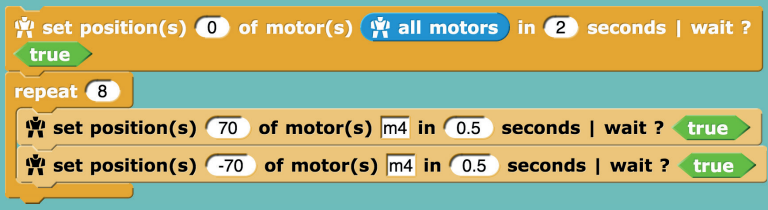
This time select *title text:*

C Add an entry for the change or alter the user *duration* movement : 

D Using a block of operators category (green): add an entry *amplitude* to allow the user to change or alter the positions of the engines during the execution of the movement *Kidinon:*

E Ez modified data block you just created to reproduce the equivalent of these two movements:





For further...

i You can choose the accepted data type for each *Entrance.* For example for the block:

It is only possible to enter res chiff in oval entry areas.

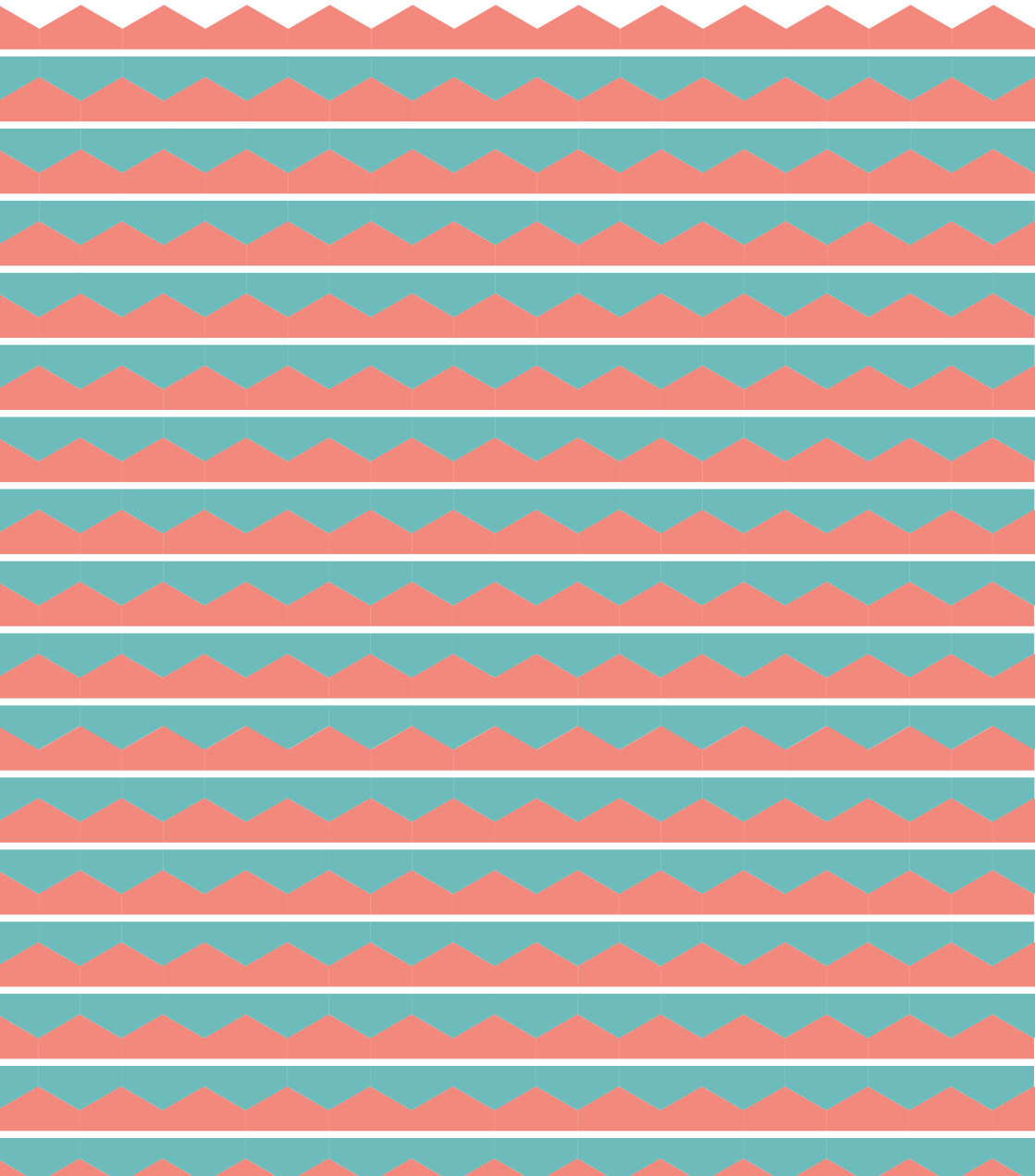
AT Click the arrow to the right of the area editing entries and explore the options:

1. Ez modified the text box and only accept numbers

2. Specify a default

3. Right-click an entry and select the type *options:* what is it for ?

B Document your block (*right click* in the block editor> *how):* that the text was ffi che in using the block.



Part 5:

If then… !

A conditional branch Type If ... Then ... allows to execute instructions according to whether a given condition is true or not.

Condition True, false condition

the blocks *report (* rounded blocks) may **refer diff erent values (** for example the block  returns a number).

the blocks *predicates (* form of blocks pointed like this one: ) **always return a value** or (Known value *Boolean).*

In a logical expression (also called Boolean), there are only two alternatives; is *True (True)* or be *False (False).* **For example :**

Can be combined:

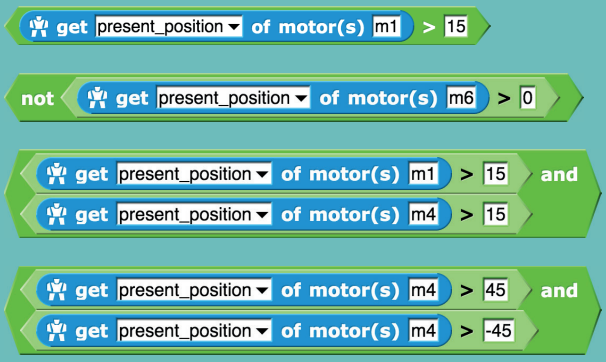
The a ffi rmation "5 + 2 = 7" is true and a ffi rmation "10 + 2 = 7" is therefore false to a ffi rmation complete false.

The a ffi rmation "3 + 2 = 5" is true and a ffi rmation "10 + 2 = 12" is true so the a ffi rmation complete is true.

It's your turn !

A Say, without using Snap !, if logical expressions below are true or false then build the blocks to inspect to your answers.

B For each logical expressions below, handle your robot to ensure that expressions are true (true) then ensure that they become false (false).



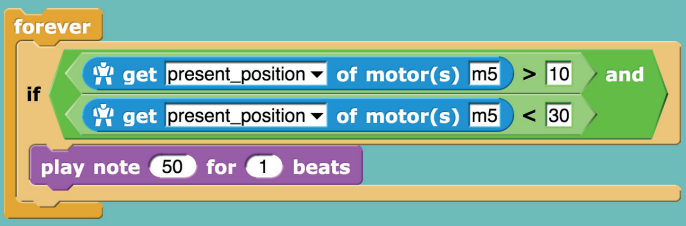
Feel free to create more!

Turn your robot musical instrument!

To play a music according to diff erent positions of the robot, we'll create a script using the concept of condition.

It's your turn !

A Create the following script and run it.



(Continued on next page)

1. Change the position of the M5 engine (manually or with Snap!) And observe what happens.

2. Ez modified the script to make sure to play a note from "60 for 0.5 beats" if the motor M4 is in a position between 0 and 60 degrees.

B Using the block 

create a script to do so to play a musical note if all engines are in a position between -5 and 5 degrees.

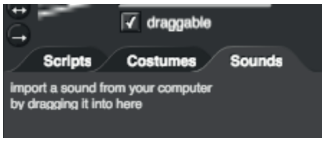
i It is also possible to encase the blocks 

to create Nested conditions.

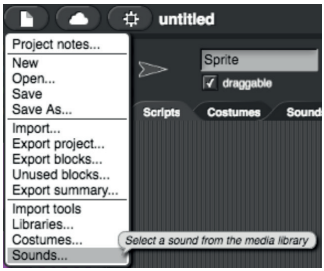
For further...

Explore the blocks in the "Sound" with what you have learned previously.

i To use a file saved in the computer, make "drag / drop" of a file of music in the "Sounds".



There is also the default sounds he proposed be opened by clicking File > Sounds.

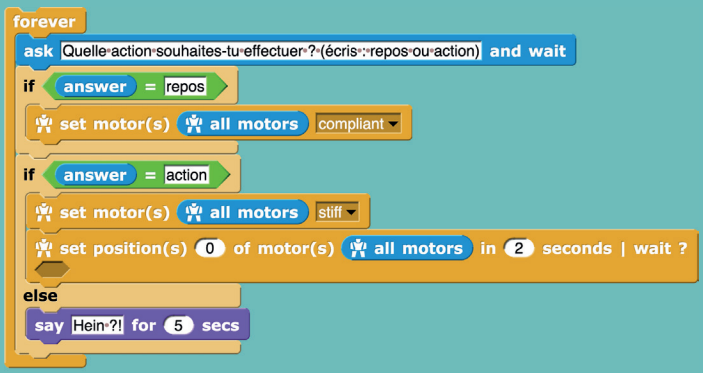


Turn your robot musical instrument!

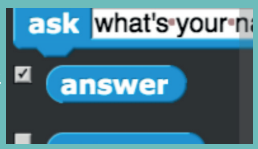
We will now use another example to understand the block

It's your turn !

A Create the following script line and execute it:

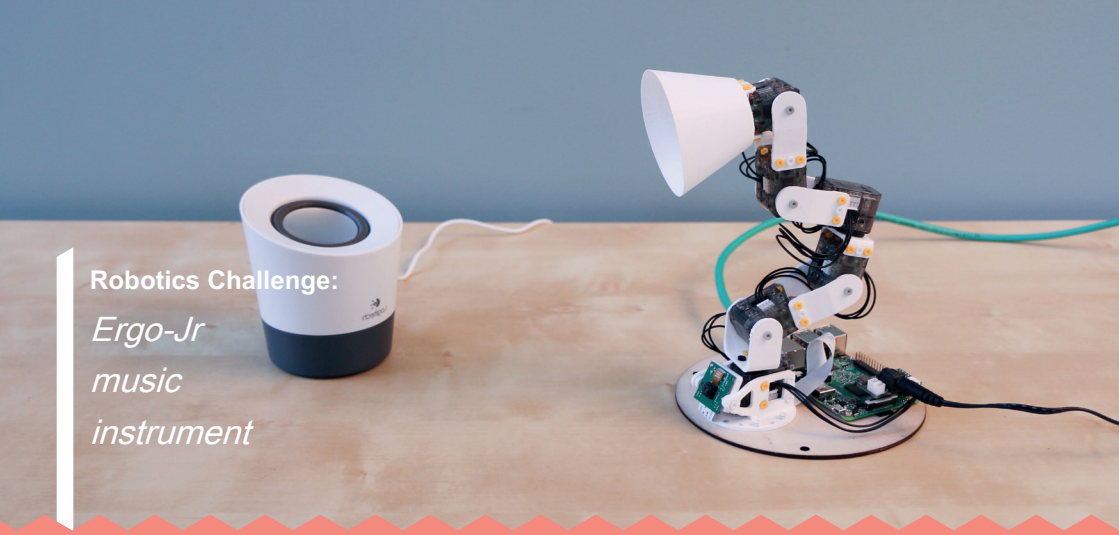


You can check the box *answer* located right in the category *sensing (* light blue) to see appear, left, real-time block value *answer.*



B What does the script? In which case he returns *Huh?! ?*

C **In the script above, the user response (*rest* or *action)***
may cause two actions diff erent. Ez modified the script to complete the list of possible actions (eg: hello, dance, High Five (high five), etc.) and create behaviors that go with it.



Robotics Challenge:

Ergo-Jr music instrument

Goal:

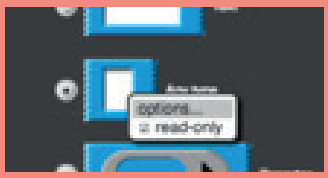
Use Poppy Ergo like a musical instrument; depending on the position of the motors it can play music diff erent. As constraints, you can:

- Create interacting with the user (block 

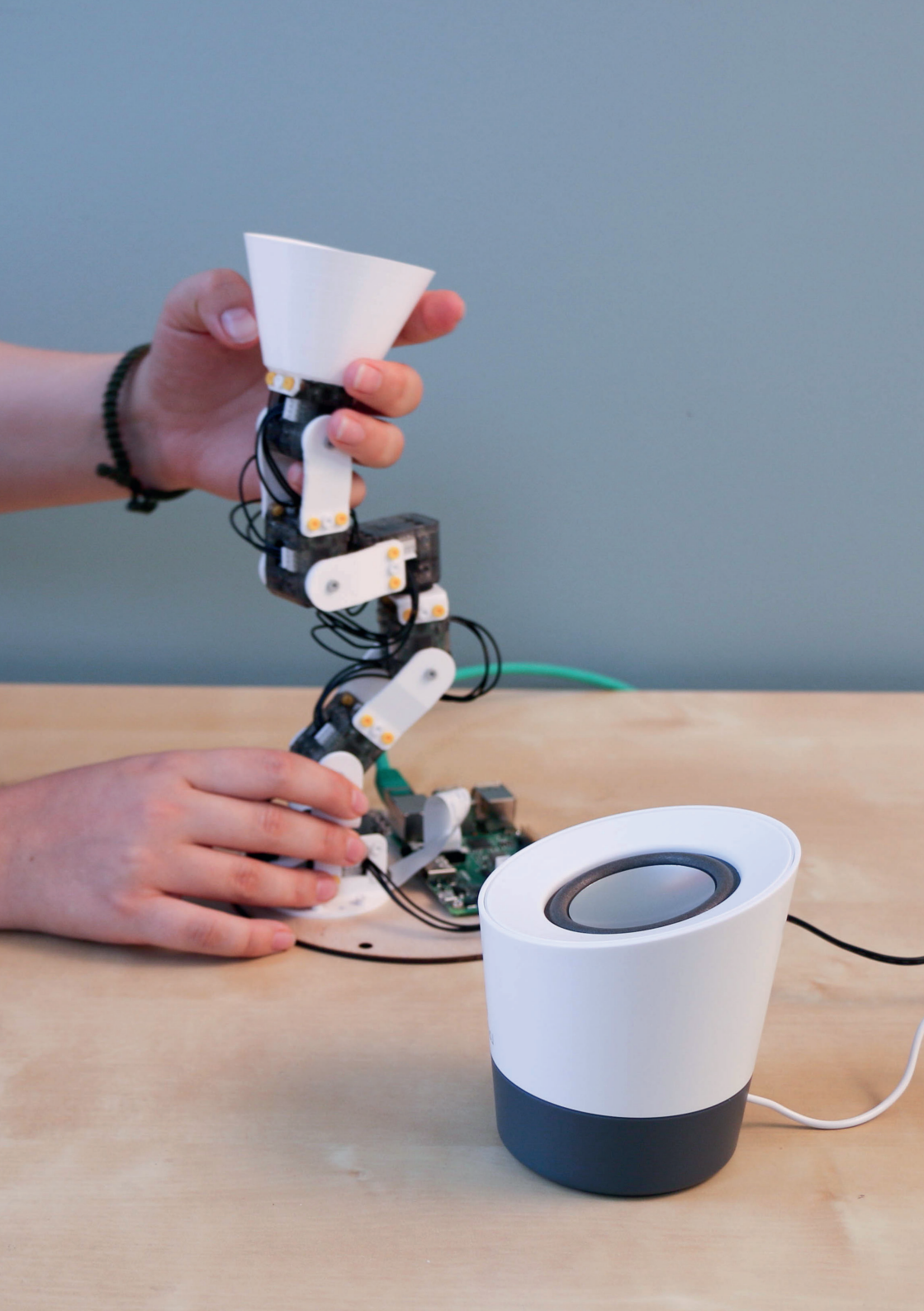
)
- Create a music box where you can select:
  - music note: C (C), D (D), E (E), F (f), Sol (g) The (A) If (B), C (C)

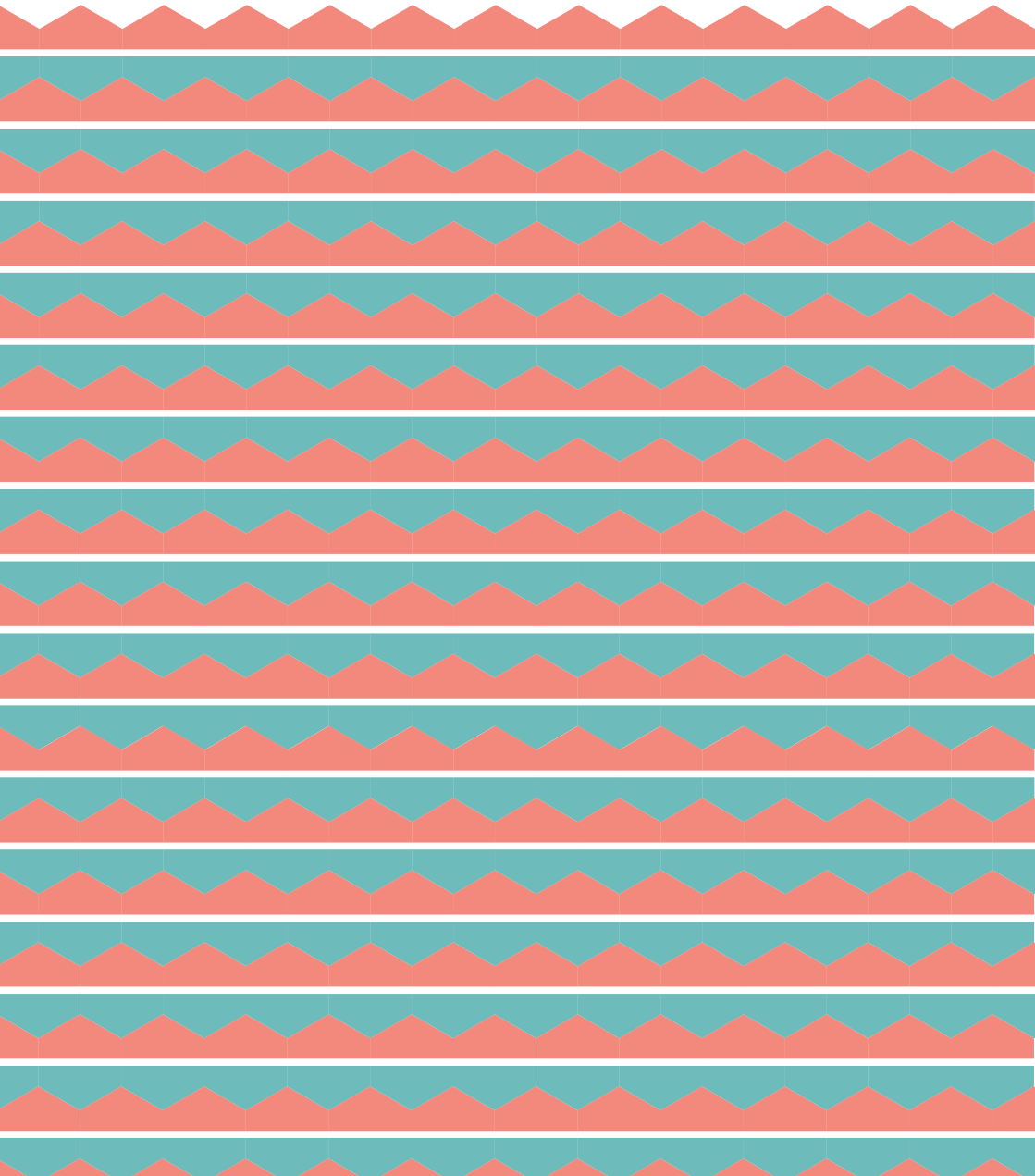
i To create default options, as this block:

You must edit the input box and then
right-click an entry and select the type *options*



- The length (black, white, etc.)
- The octave (optional)





Part 6:

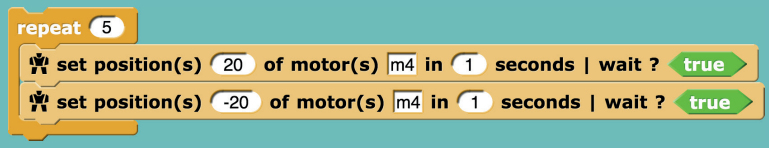
*The block **for***

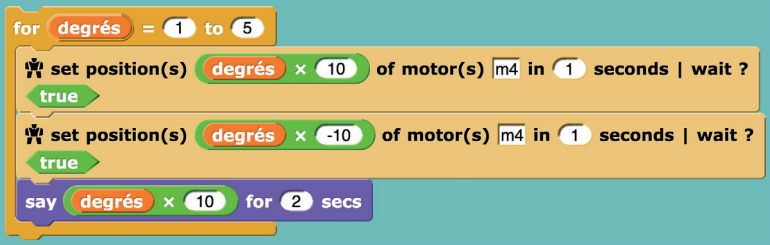
The block 

is a case especially loops that we saw in Part 3 (use repetition). It is very convenient to repeat instructions using a value that is changing.

It's your turn !

A Create and run both scripts below.

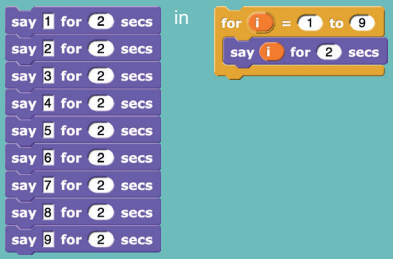




1. What diff erence main do you see between the two scripts?

2. What is the value of the variable *degrees* for each step of the loop?

i The block *for* enables simplified er a long script. Such as :



B The modified ez *for block* the second script: Replace *1 to 5* by *5 to 1.* What is going on ?

C Ez modified the script to the engine *m4* changes position in steps of 20 degrees: 20 -20 and 40 -40 and 60 -60 and fi nally 80 and -80.

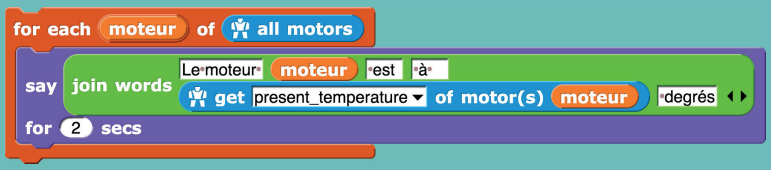
D Complete with the new movement, not always by 20 degrees: 80 -80 and 60 -60 to 10 -10 etc.

A little more complicated: the block

allows for action on each element (item) of a list.

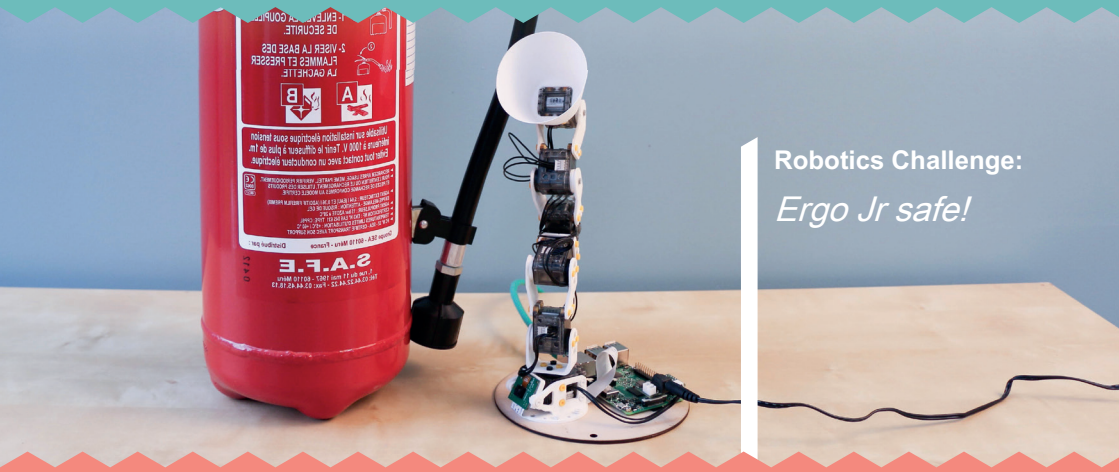
It's your turn !

Observe the following script without registration and try to guess what it does:



1. Create and test the script to inspect to your previous intuition.

2. Ez modified the script to replace the list  with a list including engines m2, m3, M5 and M6.



Robotics Challenge:

Ergo Jr safe!

i

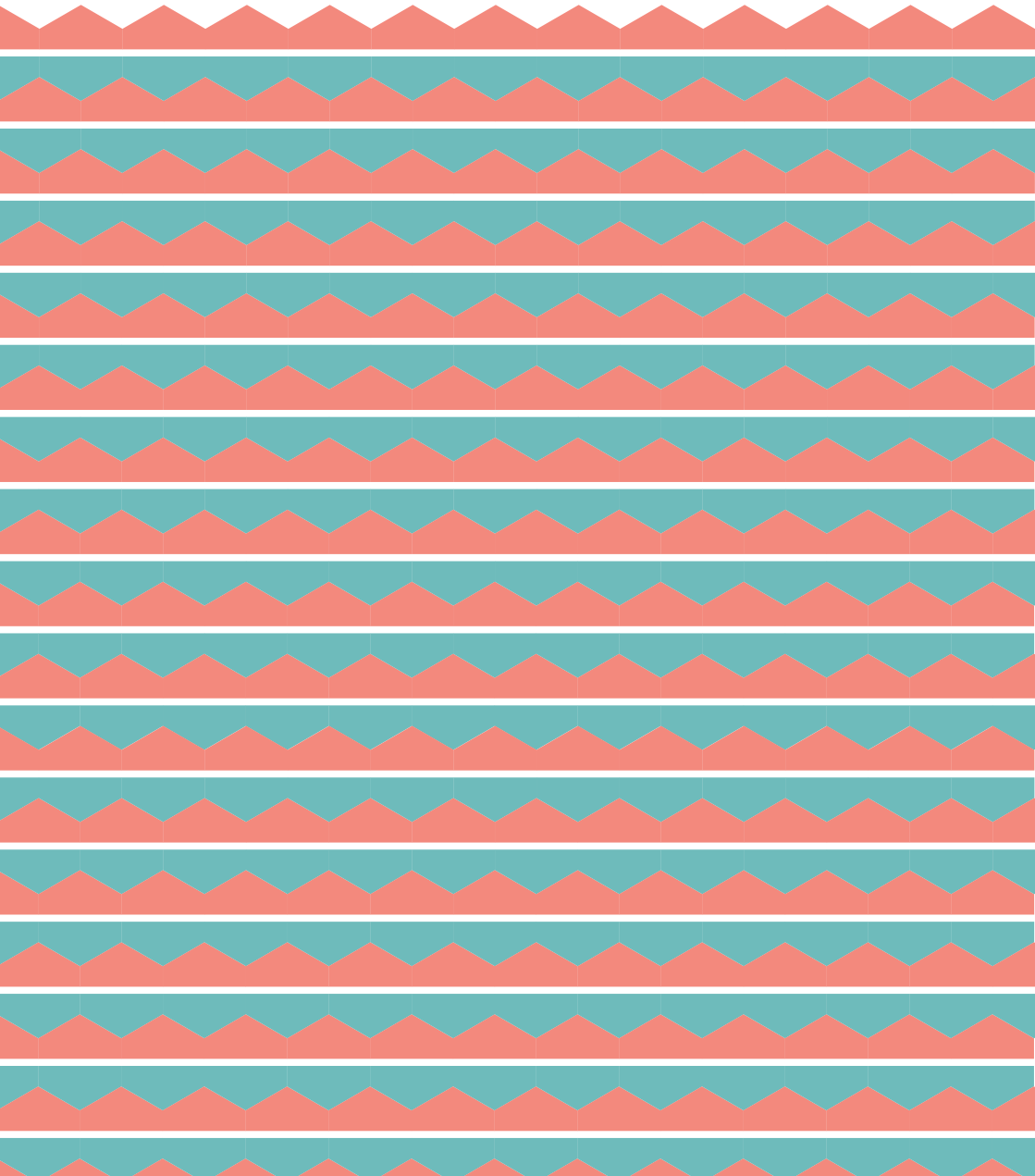
Goal:

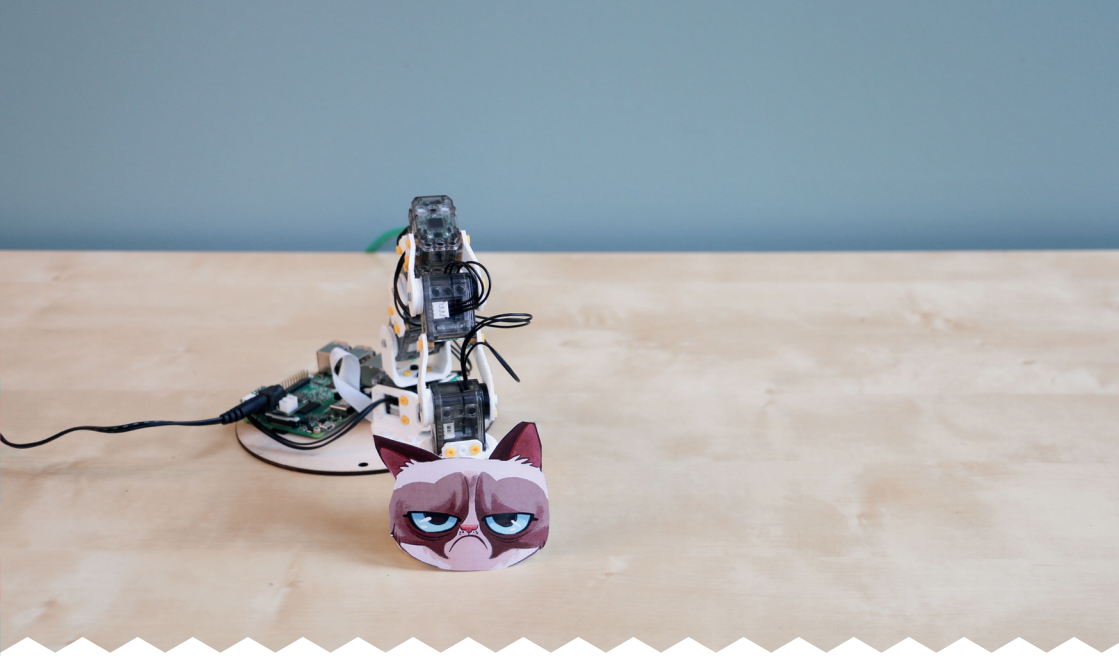
can even be damaged. The block allows to know the M1 engine temperature.

- Create by Snap! the alarm of your choice (sound, behavior etc.) that is triggered when the increase. When the temperature of an engine is too high, it is in error (red LED blinking) and temperature of an engine is too high, before it snaps into safety (red LED blinking).

Depending on the position of the robot and behaviors that eff ectue, engine temperature can

- Experiment to estimate what a reasonable trigger temperature for your Poppy Ergo Jr.





Part 7:

The variables

You have :

Created a variable as an input to a block,

Used Variable 

to store and reuse the user response, the manipulated variable block gives you,

You can create additional variables to include in your script with the block

It's your turn !

A Create, name a variable and then give it a value:

For that :

1. Create a variable with the block 

2. Name the variable by clicking 

3. Give a value to the variable using the block 

The drop-down menu to select the variable.

B Create the following script and run it several times to try:

You can use the block 

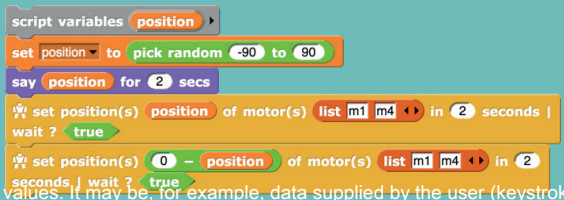
ffi expensive for a value a variable.

C Look at the two scripts below, try to guess what the diff érencie and verifi ez it by executing turns.

Now use the value generated by the variable of the script on your robot.

It's your turn !

A Observe the following script and try to guess what it does:



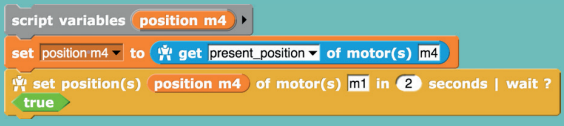
program. They say she is "variable" because the value can change.

values. It may be, for example, data supplied by the user (keystrokes) or performance of a

1. Create it and test it to inspect to.

2. Ez modified the script and add a variable *duration* and given it the value *1.5*

B Create and run the following script and see what it does: In programming, we need to store



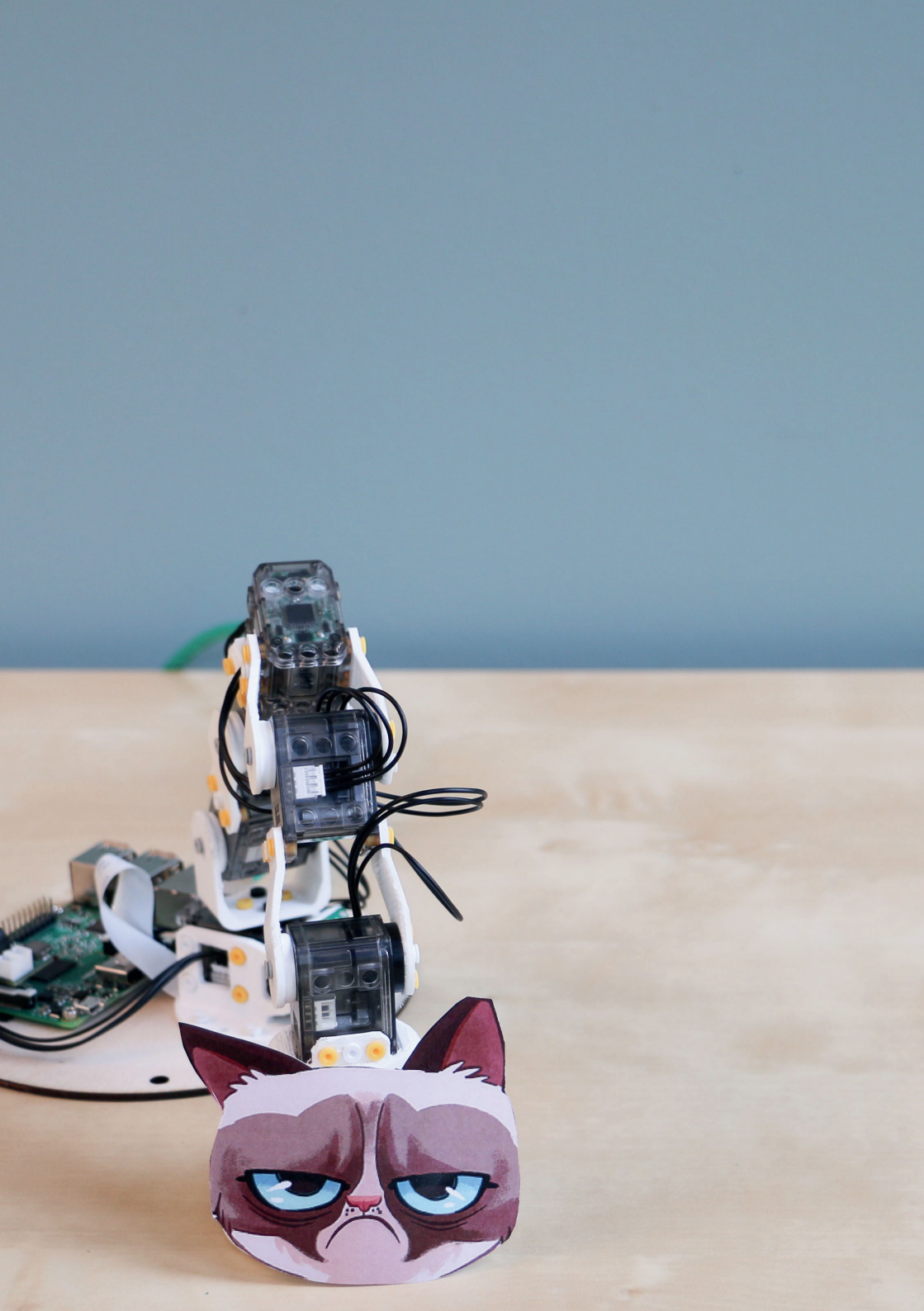
C Ez modified the script so it also gives the engine *m6* the same position as the block *m5*.

Robotics Challenge:

Ergo Jr is grumpy!

Goal:

Create a program that detects the position of the robot Ergo Jr and puts it back in its original position as soon as a person handles to change its position.



Activity Ideas

Each idea presented here is a subject in the booklet of the forum (forum.poppy-project.org: Education category), with tips, advice, exchanges with other argumentative!

Easy :

* ***Ergo Jr stage***

Create a keyboard-piano by associating a note to a key on the keyboard and a position of a motor. Thus, by pressing the keyboard, it plays both sounds to the computer and movements Ergo Jr.

* ***Ergo Jr ranks containers***

Ergo Jr program so that it grips and stacks of boxes (containers) in a specifi area, based on the QR code ffi ket thereon. (the block

allows to select an QR code: the codes are at the end of the booklet).

* ***Ergo Jr made interesting!***

Create and decorate positions / disguise Ergo Jr to illustrate phrases related feelings or behavior and make gifs / videos / images.

For example: "When I'm scared ... I'm hiding," "When I'm happy I'm the clown."

variations:

- Involve QR codes with images representing, make a gif / video
- This can also be done with the behavior of everyday life "when a customer is angry ... I remain calm and smile! ".

* ***Ergo Jr. at the Movies!***

Use Ergo Jr to control virtual objects available on the scene Snap !. For example, program Ergo Jr to make it move a sprite or draw something on the stage in the handling.

* ***Ergo Jr is Explorer!***

Using the base engine in motor mode and those of the end sensor mode (making them compliant). It is thus possible reconnaissance missions where you have to program the robot afi n he explores around him and when he is hit in a direction it focuses its exploration in that direction, or as soon as the key he made a quick withdrawal movement, etc ...

* ***Ergo Sumo Jr.***

Find and test strategies to push another opposing robot or an object outside of a zone.

• To enrich your program, you can assign keys to the settings (setting position of each motor, speed etc.)

• Create a program with Snap! for recording the score (and the number of trials). These results can also be used to drive behavior diff erent Ergo Jr (happy, sad ...).

⋆⋆ *Ergo Jr Waiter*

Build a program to catch a sugar and put it in a cup of coffee. constraint:

1. We would like to give the robot the order to serve to the left or right by simply pressing one of the buttons
- > or <-.
2. One sugar must be deposited into the cup. (the block

allows to know the force at time t) Variation:

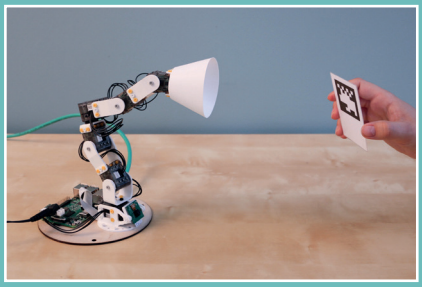
Search and grab an object. If the object is present, Ergo Jr moves elsewhere, otherwise it returns to its original position.

⋆⋆ *Ergo Jr messes ... everything!*

Check the position and speed of the robot's motors to throw the ball and drop the coconut shy (cups on top of each other).

variations:
• Throw the ball in a basket or a basket.

• Competitions between several Ergo Jr can be arranged.



⋆⋆ *Ergo Jr. invents a language*

Invent a programming language based on the use of QR codes. For this, recording a motion for each QR code, program a block for ERGO Jr performs this movement if the camera detects the QR code associated. Do the same with several QR Codes. Thus, one can play the robot choreography showing diff erent QR codes due to chain movements.

Difficult :

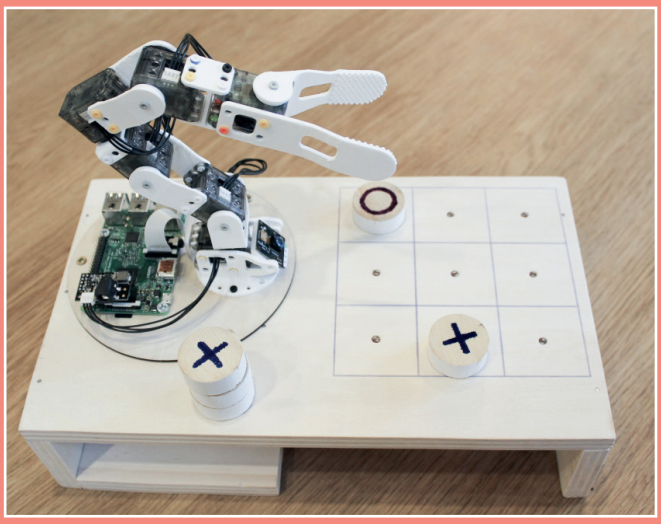


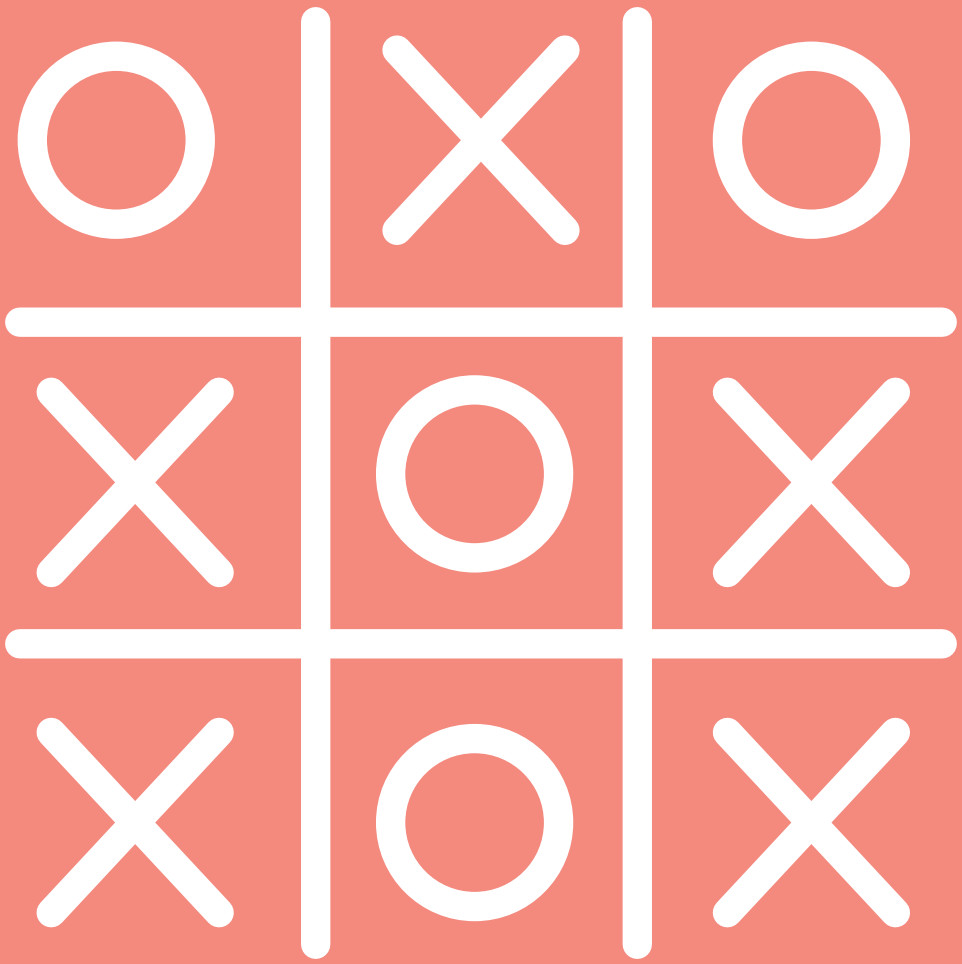
Photo by Gilles Lassus

* * * *Ergo Jr play Tic-Tac-Toe*

The game involves placing a cross in turn for a player, a circle to the other. To win, a player must align three identical symbols on the same line, the same column or the same diagonal.

To play Ergo Jr. can be programmed by demonstration so he knows reach each of the nine boxes provided for the game and draw the cross (or circle, according to the choice). The next step is to do the play by pressing a button with an algorithm to try to win.

Variant : symbols are on stacked chips ERGO Jr. picks to ask where he wants on the mat game.



Ergo Jr. • Word List *Draw is Winning*

Easy :

sun moon daisy
envelope a star one
hand a cherry cloud
hat cheese a bottle
a snail ring a
balloon banana
spider ice castle a
saucepan butterfly
car a house

Way :

truck pyramid snail
vase chair a plane a
boat a bear bag
beard a key face a
sock rabbit fork a
basket of silver leaf
mountain a look
mustache rain
coffee bread a train

Difficult :

a cat a beach a shell
clock a shoe phone a
building cow shark
computer motorcycle
duvets an egg on a flat
reindeer King bike hen
horse a lion

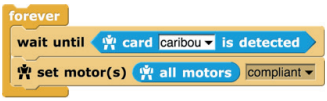
Attachments

QR codes in print

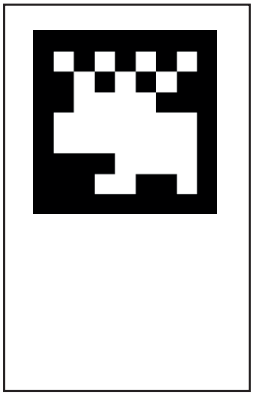
The block 

can trigger an action if the map selected is detected (by the robot camera Ergo Jr).

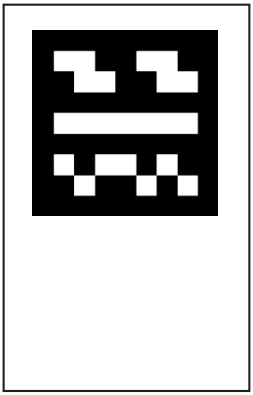
One can for example create this script:



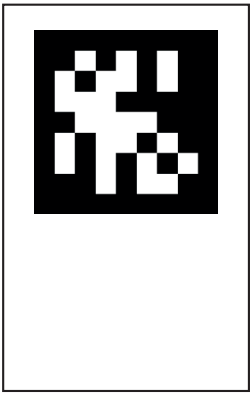
Caribou Code



Tetris code



Rabbit Code



accompanying booklet
of Poppy Ergo Jr robot

Poppy Project 2016

Version 1.0 (rev 0)