33

SMART PET

STEMJAM Teaching Guide

Developing make spaces to promote creativity around STEM in schools
Acronym: STEMJAM
Project no. 2016-1-ES01-KA201-025470

www.stemjam.eu

Co-funded by the
Erasmus+ Programme
of the European Union

SMART PET

ABSTRACT

The idea is to create an "mBot pet" which will provide a response to different human reactions and then collect information about them.

Students will code an mBot-Pet in order to use it as a data collection system. By testing different sensors, they will study the most common reactions a human being has towards a pet (such as to play with it, caress it, talk nicely or yell). The mBot-Pet will have a specific response to any of these human reactions.

When the robot has a certain number of reactions stored, the program will stop and the data collected will be showed on the screen. This information will be the starting point for creating a statistics study in which students will be able to put in practise some math's concepts (barcode elaboration, mean, median, mode, ratio, proportions…).

The second versión of this activity, consits of interact with a mBot disguised as a baby.

This second versión consists mainly of two sections.

- In the first part, a code has been developed in which the user can personalize the emotions of the baby, since through the remote control or through the Me 4 Button Actuator, by pressing a button the baby will show an emotion, and by touching another button, the baby will change the expression again. In this way, for example, small theaters can be made to call attention to the smallest people.
- The second section is that the baby will recognize 3 states, when someone or something approaches (ultrasonic sensor), if there is too much light (light sensor) and there is too much sound (sound sensor). When the baby recognizes this action, it will emit both an acoustic and luminous signal.

DIDACTIC OBJECTIVES

TECHNOLOGY

- Introduction to computational thinking.
- Study and use of different sensors.
- Assimilation, creation and programming of algorithms, to decompose complex problems into ordered sequences of simple instructions, which solve it.

MATHEMATICS

- Solving statistics and probability exercises.
- Elaboration of barcodes.
- Mean, median and mode.
- Standard deviation.
- Proportions.

STEM Subject: Science☐ Technology ☒ Engineering☒ Mathematics☒

Education Level: 12-14 years☐ 14-16 years☒

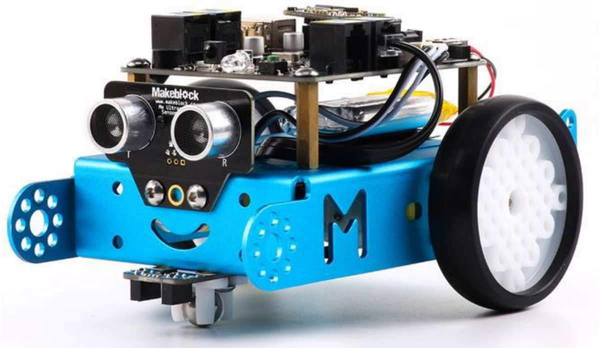
PROBLEM STATEMENT

The aim of the activity is to learn about different sensors and collect information which will be used to solve Math exercises (such as barcode elaboration and calculus of mean, median, mode, ratio, proportions…) by applying technological competences and working in a ludic environment.

This target will be achieved, since the students will have to design the programming blocks related to the interaction between the Robot-Pet and the human, as well as the different sound and dance effects resulting from the robot's responses.

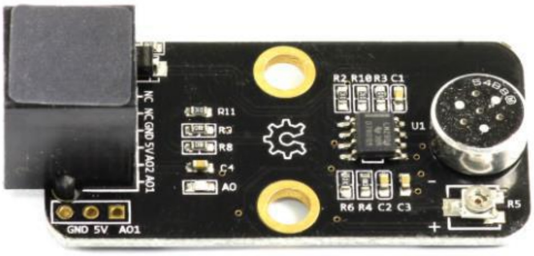
BOM (Bill of Materials Needed)

- mBot:



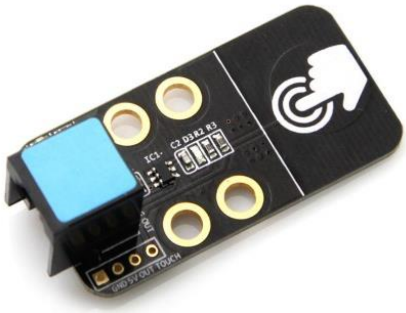
- LED matrix:

❖ Me **Sound Sensor**:

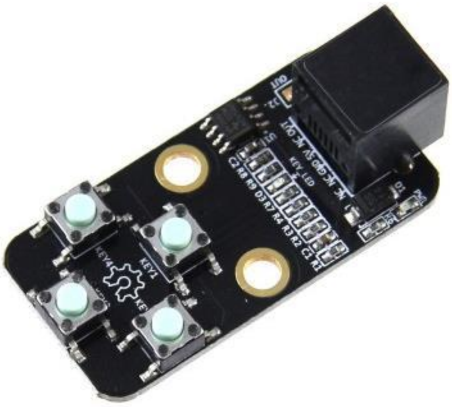


❖ Me **Ultrasonic Sensor**:

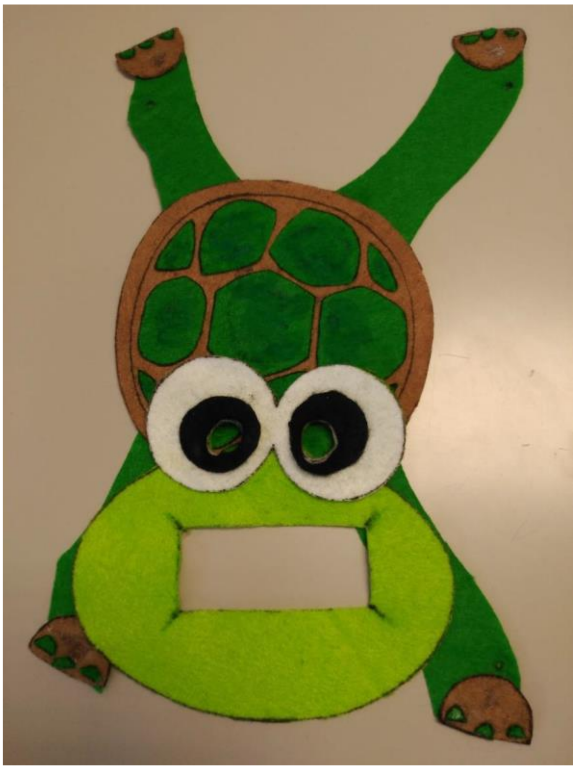
❖ Me **Touch Sensor**:



❖ Me **4 Button Actuator**:



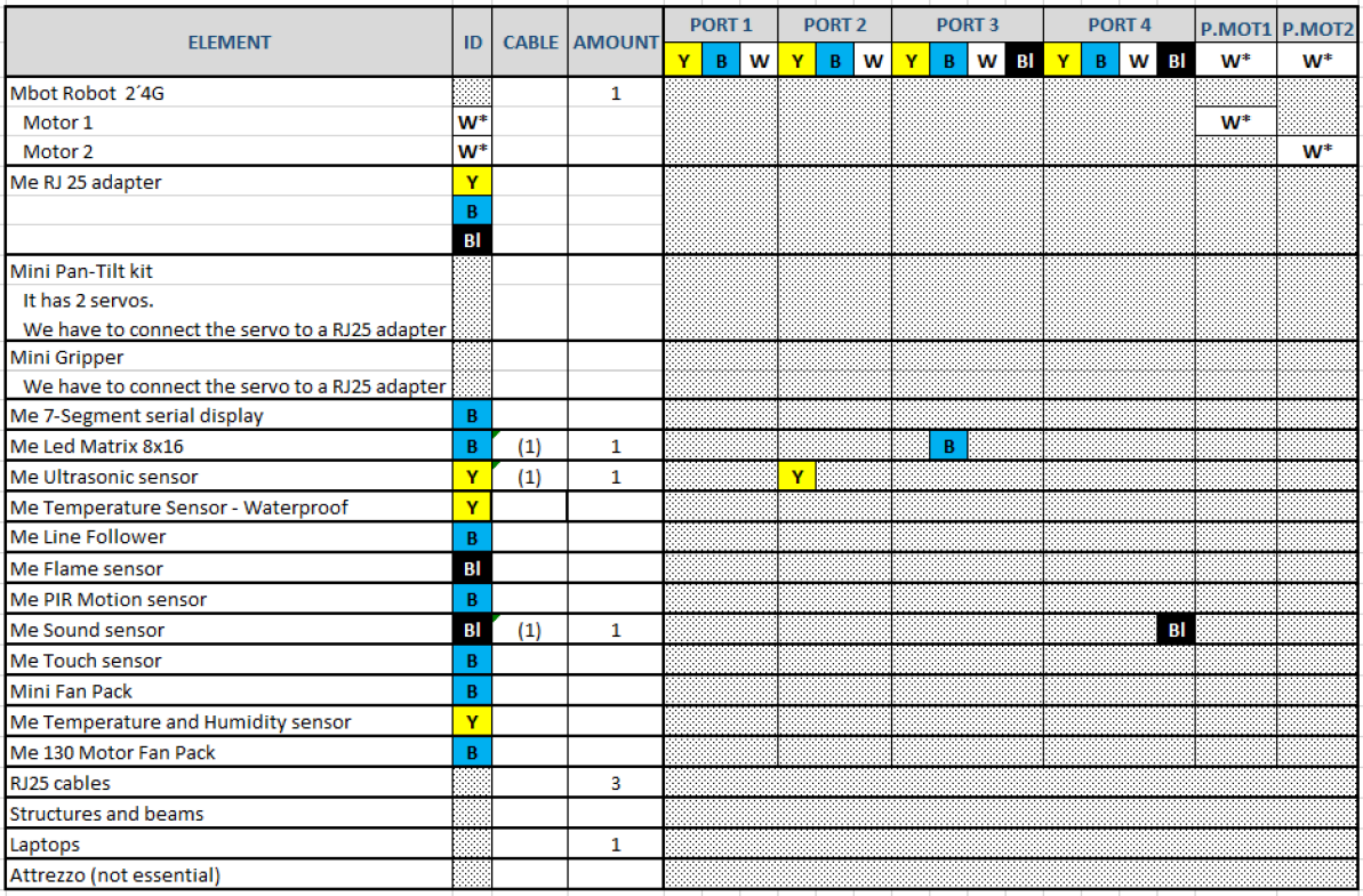
- Disguise:



- Pet costume (colour paper and cotton).
- Rest of Attrezzo (not essential).

In the following table you can see to which ports we have connected each component. The codes have been programmed following these connections.

ELEMENT	ID	CABLE	AMOUNT	PORT 1			PORT 2			PORT 3				PORT 4				P.MOT1	P.MOT2
				Y	B	W	Y	B	W	Y	B	W	Bl	Y	B	W	Bl	W*	W*
Mbot Robot 2´4G			1																
Motor 1	W*																	W*	
Motor 2	W*																		W*
Ultrasonic sensor	Y	1	1	T															
Touch sensor	B	1	1					B											
Sound sensor	Bl	1	1										Bl						
Matriz de LEDs	B	1	1												B				
RJ25 cables			4																
Structures																			
Support P1			3																
Cut-out beam			4																
Plate 45°			1																
Laptops			1																
Atrezzo (not essential)			X																



ELEMENT	ID	CABLE	AMOUNT	PORT 1			PORT 2			PORT 3				PORT 4				P.MOT1	P.MOT2
				Y	B	W	Y	B	W	Y	B	W	Bl	Y	B	W	Bl	W*	W*
Mbot Robot 2´4G			1																
Motor 1	W*																	W*	
Motor 2	W*																		W*
Me RJ 25 adapter	Y																		
	B																		
	Bl																		
Mini Pan-Tilt kit																			
It has 2 servos.																			
We have to connect the servo to a RJ25 adapter																			
Mini Gripper																			
We have to connect the servo to a RJ25 adapter																			
Me 7-Segment serial display	B																		
Me Led Matrix 8x16	B	(1)	1								B								
Me Ultrasonic sensor	Y	(1)	1				Y												
Me Temperature Sensor - Waterproof	Y																		
Me Line Follower	B																		
Me Flame sensor	Bl																		
Me PIR Motion sensor	B																		
Me Sound sensor	Bl	(1)	1														Bl		
Me Touch sensor	B																		
Mini Fan Pack	B																		
Me Temperature and Humidity sensor	Y																		
Me 130 Motor Fan Pack	B																		
RJ25 cables			3																
Structures and beams																			
Laptops			1																
Attrezzo (not essential)																			

ACTIVITY DESCRIPTION

First version

The *mBot-Pet* stays still and waits for a human to appear. Once it detects the human, it goes running towards him/her in order to interact with this person. From this point on, the program depends on the human behaviour:

- If the human touches the *mBot-Pet*, it shows its love with harts, happy sounds and intermittent pink lights and "Caress" variable adds a point.

- If the human plays with the *mBot-Pet*, it plays back and "Playing" variable adds a point.

- If the human talks to the *mBot-Pet* normally, it communicates back by showing a heart and "Nice words" variable adds a point.

- If the human talks to the *mBot-Pet* loudly, it communicates back by showing an unexpected and surprise message and "Bad words" variable adds a point.

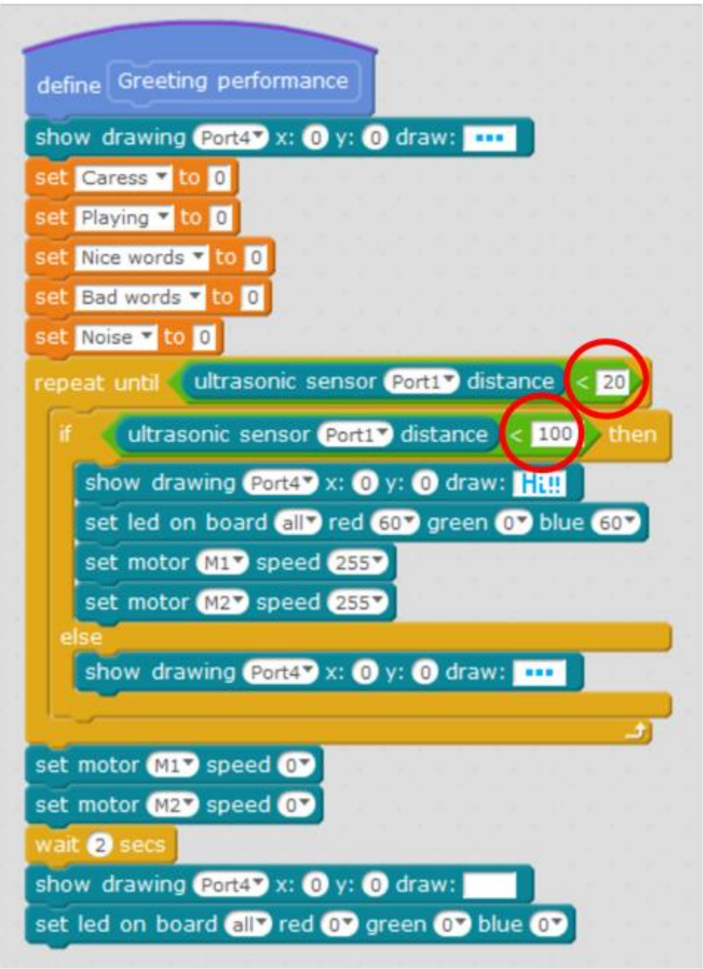
The robot will interact with the human until it has a specific number of responses and then the program will end by showing the data collected. There has been settled a number of 10 responses, but this value can be modified.

After downloading Mblock software, it will be paired with MBot by using the 2.4G Wireless Serial Port. Once the matching is done, the next step to start with PROGRAMMING tasks.

1. GREETING DEFINITION

The mBot stays still until it detects a presence at a distance of 100 cm. Then, it comes to the detected presence and stops 20 cm before meeting it, while showing "Hi!!" message and turning on pink lights on board.

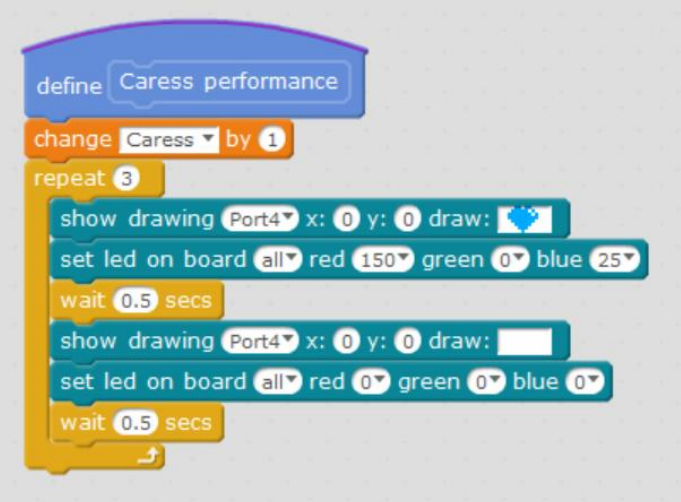
This two distances can be editable, depending on the specifications needed.



```
define Greeting performance
show drawing Port4 x: 0 y: 0 draw:
set Caress to 0
set Playing to 0
set Nice words to 0
set Bad words to 0
set Noise to 0
repeat until ultrasonic sensor Port1 distance < 20
    if ultrasonic sensor Port1 distance < 100 then
        show drawing Port4 x: 0 y: 0 draw: Hi!!
        set led on board all red 60 green 0 blue 60
        set motor M1 speed 255
        set motor M2 speed 255
    else
        show drawing Port4 x: 0 y: 0 draw:
set motor M1 speed 0
set motor M2 speed 0
wait 2 secs
show drawing Port4 x: 0 y: 0 draw:
set led on board all red 0 green 0 blue 0
```

2. <u>CARESS PERFORMANCE DEFINITION:</u>

Caress performance consists on adding one point to variable “Caress”, and then repeating three times a combined intermittent message about hearts and pink lights on board.

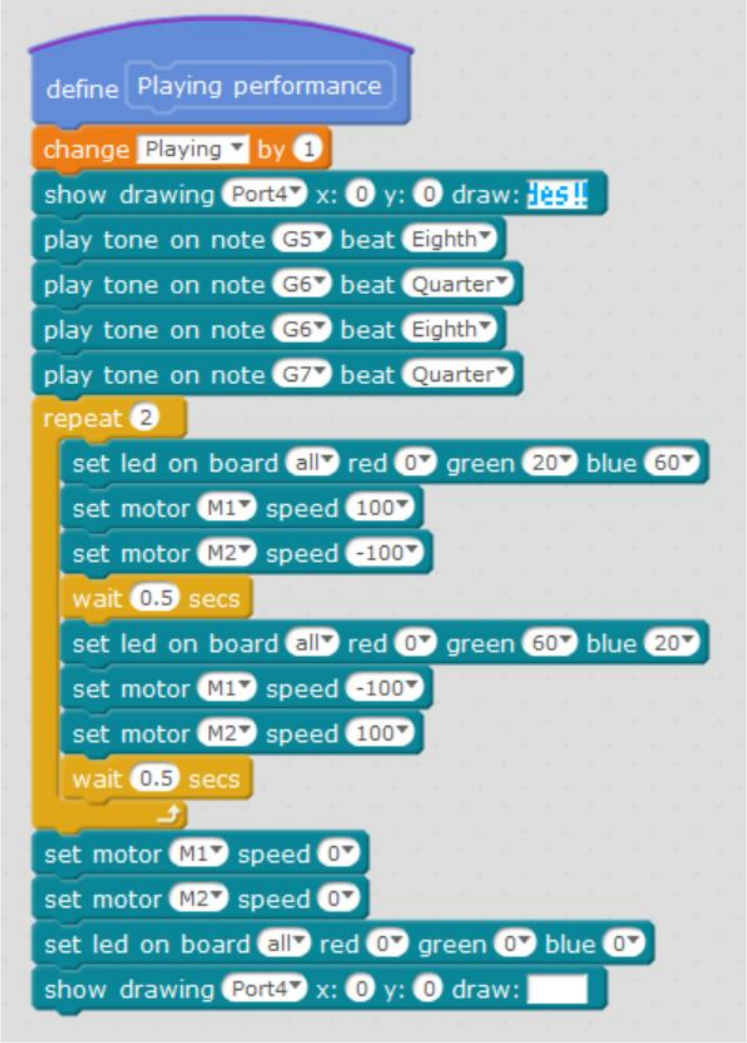


```
define Caress performance
change Caress by 1
repeat 3
    show drawing Port4 x: 0 y: 0 draw:
    set led on board all red 150 green 0 blue 25
    wait 0.5 secs
    show drawing Port4 x: 0 y: 0 draw:
    set led on board all red 0 green 0 blue 0
    wait 0.5 secs
```

3. PLAYING PERFORMANCE DEFINITION:

Playing performance consists on adding one point to variable “Playing”, and then showing the following effects:

- Message “yess!!”
- Short melody
- Leds and motion series repeated twice
- Clearing all the previous



```
define Playing performance
change Playing by 1
show drawing Port4 x: 0 y: 0 draw:
play tone on note G5 beat Eighth
play tone on note G6 beat Quarter
play tone on note G6 beat Eighth
play tone on note G7 beat Quarter
repeat 2
    set led on board all red 0 green 20 blue 60
    set motor M1 speed 100
    set motor M2 speed -100
    wait 0.5 secs
    set led on board all red 0 green 60 blue 20
    set motor M1 speed -100
    set motor M2 speed 100
    wait 0.5 secs
set motor M1 speed 0
set motor M2 speed 0
set led on board all red 0 green 0 blue 0
show drawing Port4 x: 0 y: 0 draw:
```

4. COMMUNICATION PERFORMANCE DEFINITION:

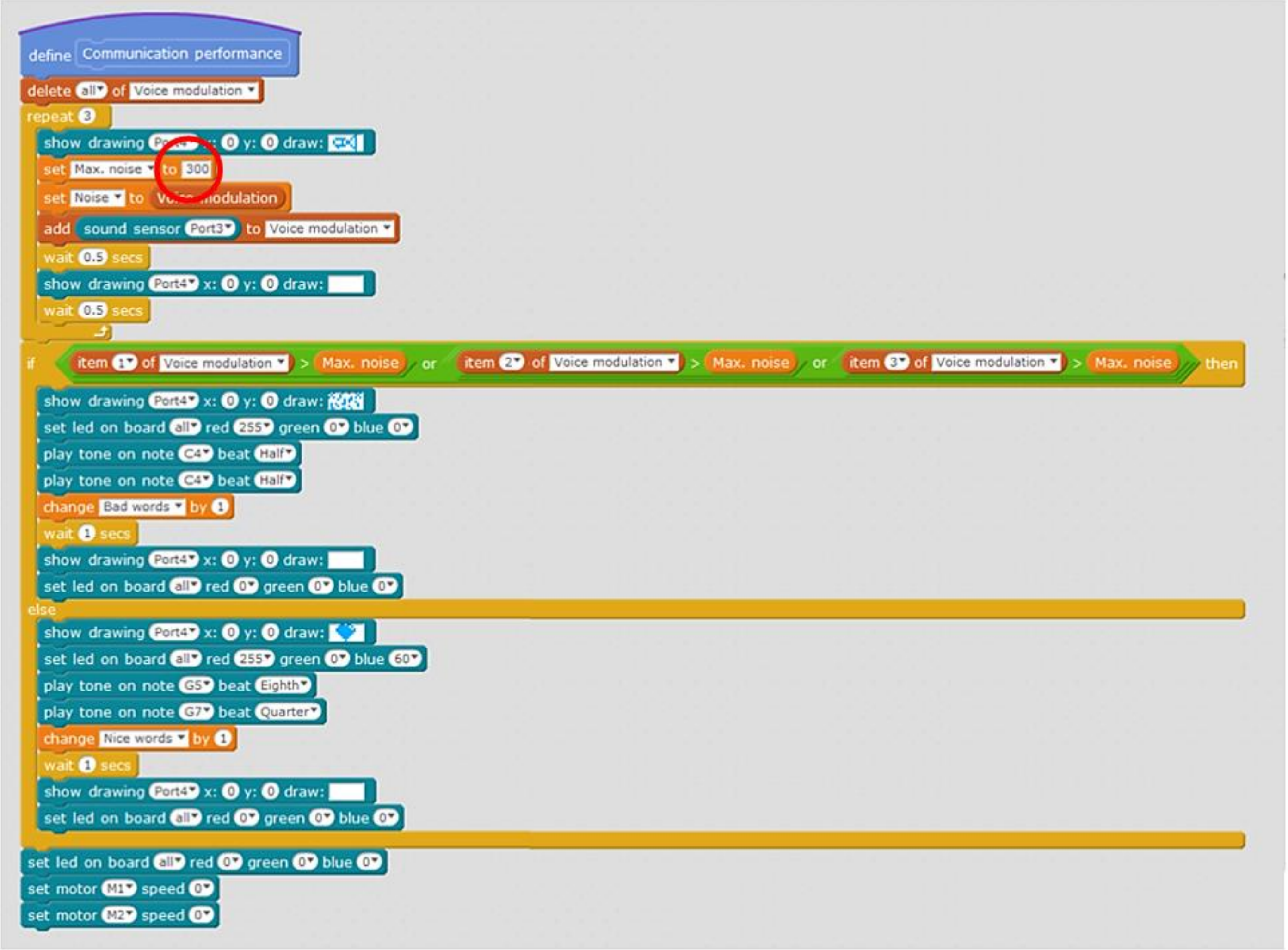
Communication performance has been designed by taking into account the voice modulation of a human being. According to this, when a human being is angry he/she usually speaks in a higher voice tone and communicates short and direct messages.

In order to elaborate this code, there are needed two phases:

- Phase 1: “Voice modulation” list is cleared. Sound sensor collects three different sound values in three seconds, and they are stored on “Voice modulation” list. The variable “Max. Noise” is set to a value which is considered as higher than normal (300 in our case, but this value can change depending on the physic conditions of the environment).

- Phase 2: If one of the three collected values is higher than the value set as “Max. noise”, the mBot perceives yells and it reacts by showing a confused message on the Led display, red lights on board and low music tones. Then, “Bad words” value adds a point.

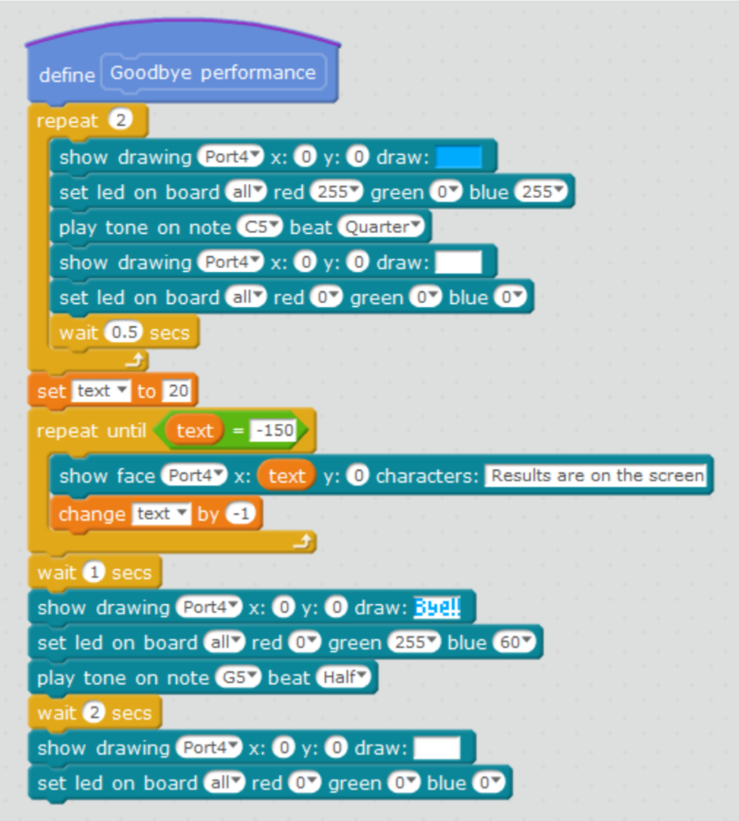
On the contrary, if all the values collected are below the value set as “Max. noise”, the mBot perceives communication with the human being and reacts showing a loving message of hearts and pink lights on board. Then, “Nice words” value adds a point.



```
define Communication performance
delete all of Voice modulation
repeat 3
    show drawing Port4 x: 0 y: 0 draw:
    set Max. noise to 300
    set Noise to Voice modulation
    add sound sensor Port3 to Voice modulation
    wait 0.5 secs
    show drawing Port4 x: 0 y: 0 draw:
    wait 0.5 secs
if item 1 of Voice modulation > Max. noise or item 2 of Voice modulation > Max. noise or item 3 of Voice modulation > Max. noise then
    show drawing Port4 x: 0 y: 0 draw:
    set led on board all red 255 green 0 blue 0
    play tone on note C4 beat Half
    play tone on note C4 beat Half
    change Bad words by 1
    wait 1 secs
    show drawing Port4 x: 0 y: 0 draw:
    set led on board all red 0 green 0 blue 0
else
    show drawing Port4 x: 0 y: 0 draw:
    set led on board all red 255 green 0 blue 60
    play tone on note G5 beat Eighth
    play tone on note G7 beat Quarter
    change Nice words by 1
    wait 1 secs
    show drawing Port4 x: 0 y: 0 draw:
    set led on board all red 0 green 0 blue 0
set led on board all red 0 green 0 blue 0
set motor M1 speed 0
set motor M2 speed 0
```

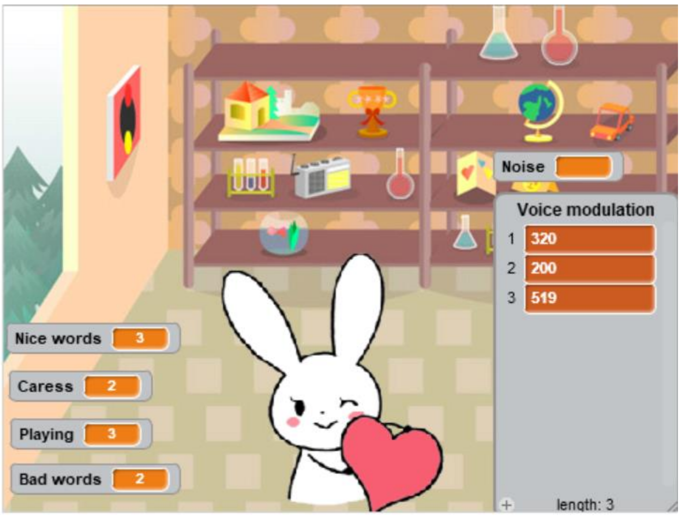
5. GOOGBYE PERFORMANCE

Goodbye performance consists on adding one point to variable “Caress”, and then repeating twice a combined intermittent message about illuminating entirely the Led display and showing lights on board while playing a tone.
Then, the mBot communicates that results taken on the activity are available on the laptop screen, in mBlock program.
Finally, it says a “Bye!!” message while showing lights on board and playing a tone, and everything is cleared.



```
define Goodbye performance
repeat 2
    show drawing Port4 x: 0 y: 0 draw:
    set led on board all red 255 green 0 blue 255
    play tone on note C5 beat Quarter
    show drawing Port4 x: 0 y: 0 draw:
    set led on board all red 0 green 0 blue 0
    wait 0.5 secs
set text to 20
repeat until text = -150
    show face Port4 x: text y: 0 characters: Results are on the screen
    change text by -1
wait 1 secs
show drawing Port4 x: 0 y: 0 draw: Bye!!
set led on board all red 0 green 255 blue 60
play tone on note G5 beat Half
wait 2 secs
show drawing Port4 x: 0 y: 0 draw:
set led on board all red 0 green 0 blue 0
```

mBlock screen’s interface will show how the variables increase depending on the interaction with the human, just as the picture underneath.



Noise

Voice modulation

1	320
2	200
3	519

length: 3

Nice words 3

Caress 2

Playing 3

Bad words 2

6. MAIN PROGRAMME: the sequence of the activity

The main program of the activity is showed on the following picture:

- Firs of all “Greeting performance” is done.

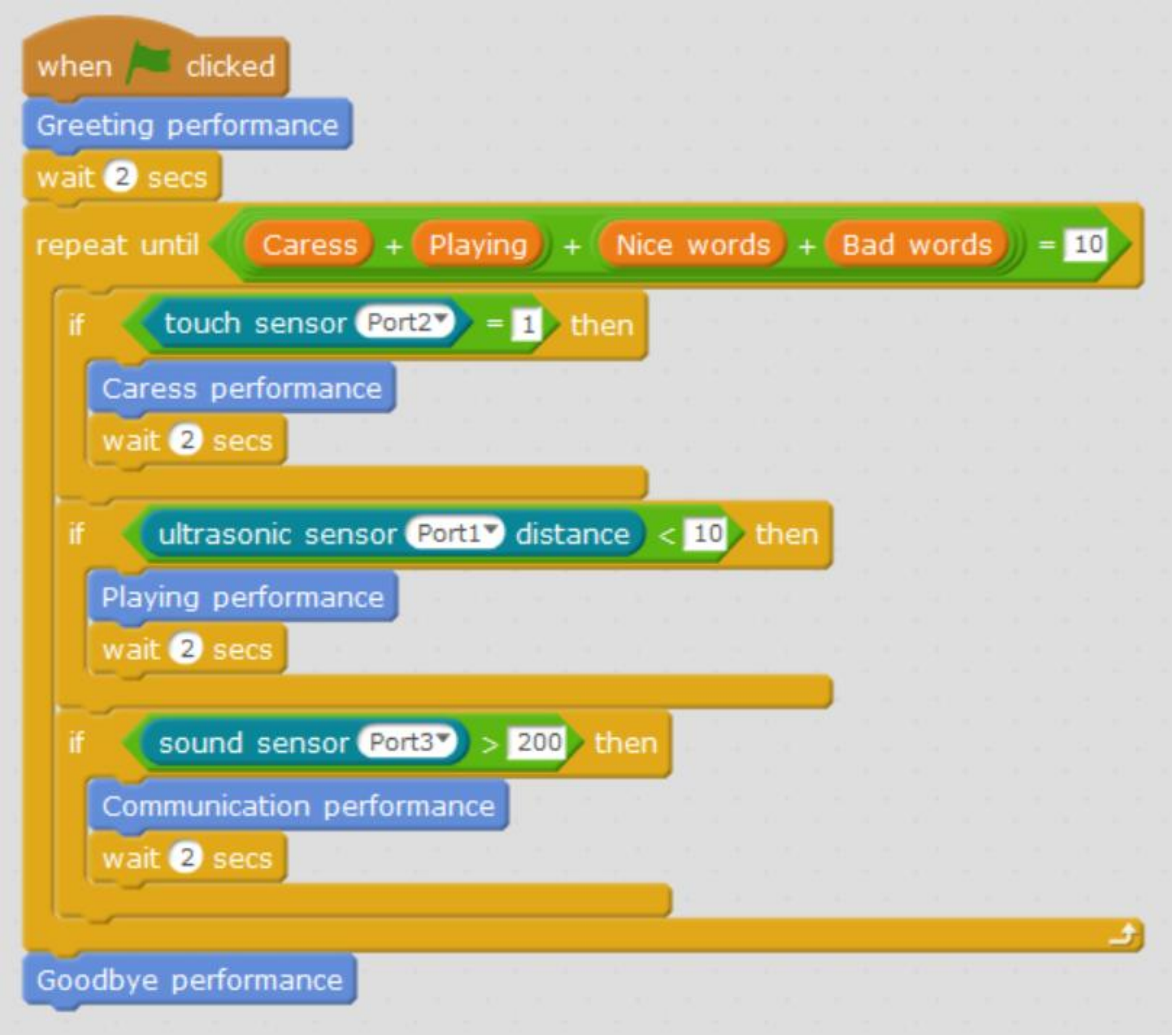
- Then, the mBot may follow three different paths depending on the interaction with the human:

  1. If the person touches it, it will complete “Caress performance”.

  2. If the person detects someone to a close distance (10 cm) it, it will complete “Playing performance”.

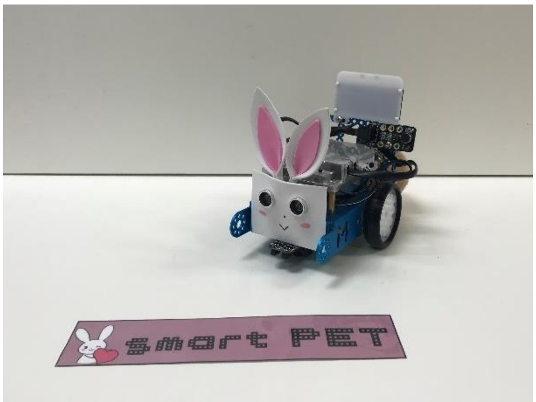
  3. If the person starts talking at a normal tone voice (detected as “200” value), it will complete “Communication performance”.

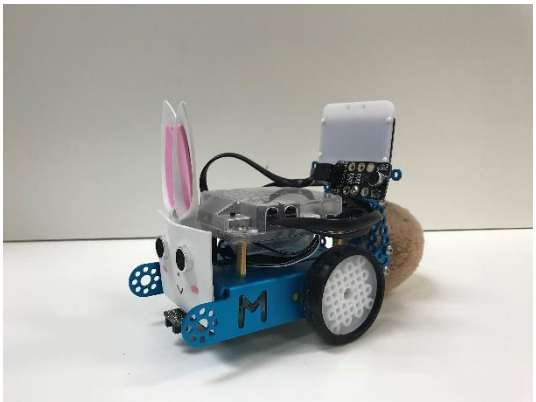
- When it has collected a specific number of responses (in our case 10 responses, but this quantity is editable), “Goodbye performance” will be done and the programme will end. The results will be shown on the screen.

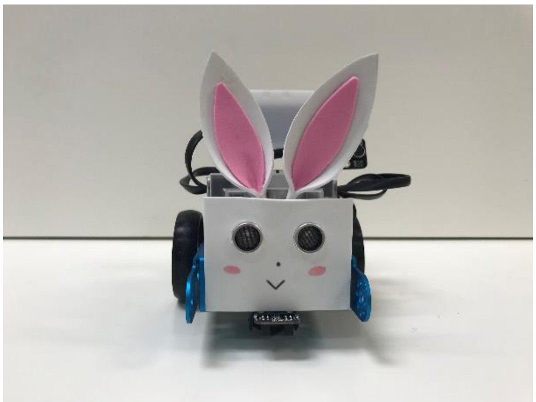


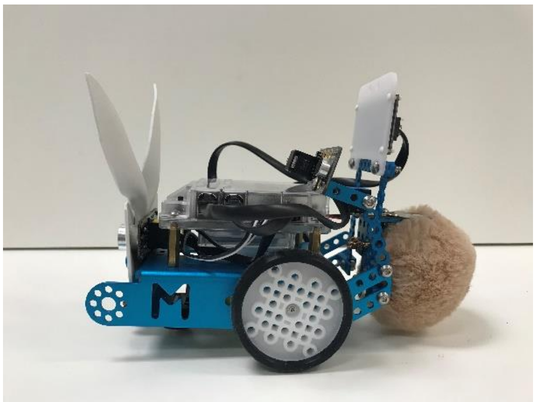
```
when 🏴 clicked
Greeting performance
wait 2 secs
repeat until ((Caress + Playing) + Nice words) + Bad words) = 10
    if touch sensor Port2▾ = 1 then
        Caress performance
        wait 2 secs
    if ultrasonic sensor Port1▾ distance < 10 then
        Playing performance
        wait 2 secs
    if sound sensor Port3▾ > 200 then
        Communication performance
        wait 2 secs
Goodbye performance
```

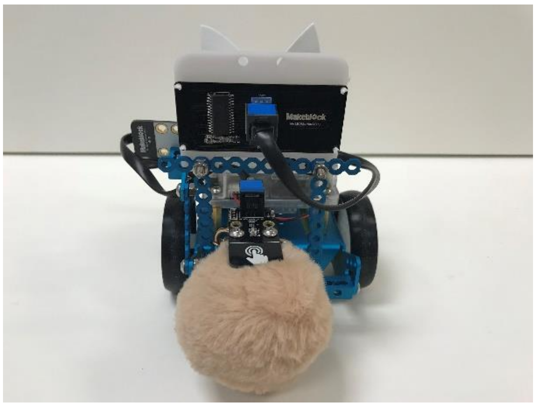
Once, the programming is finished, we start BUILDING UP THE STRUCTURE where all the mechanical elements will be set, just as the electronic elements.

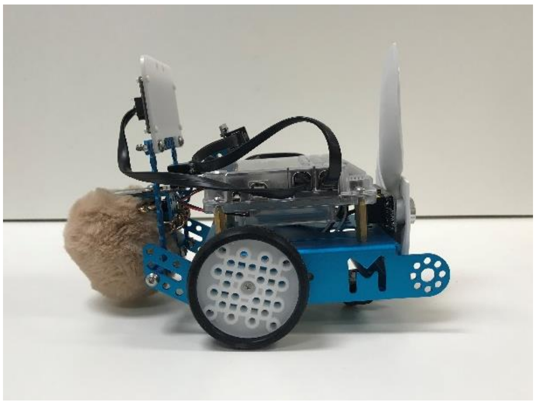






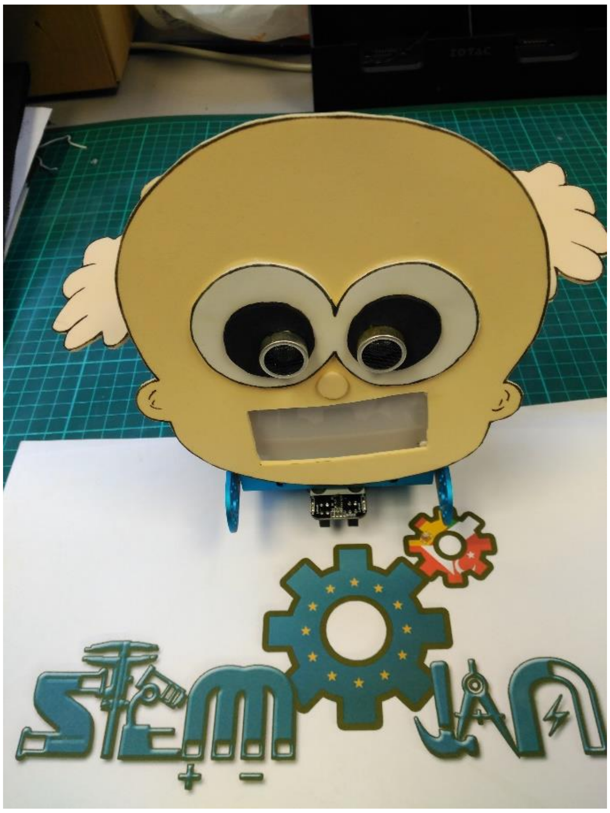


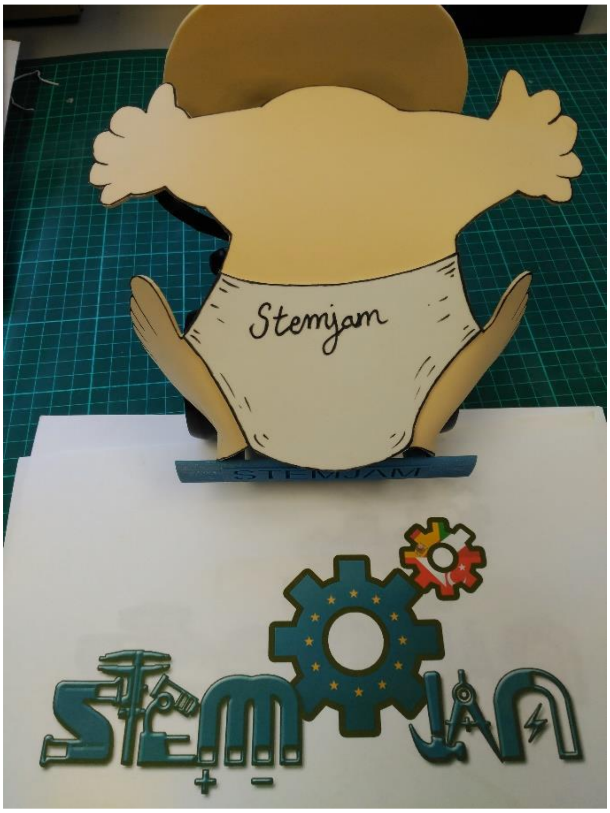


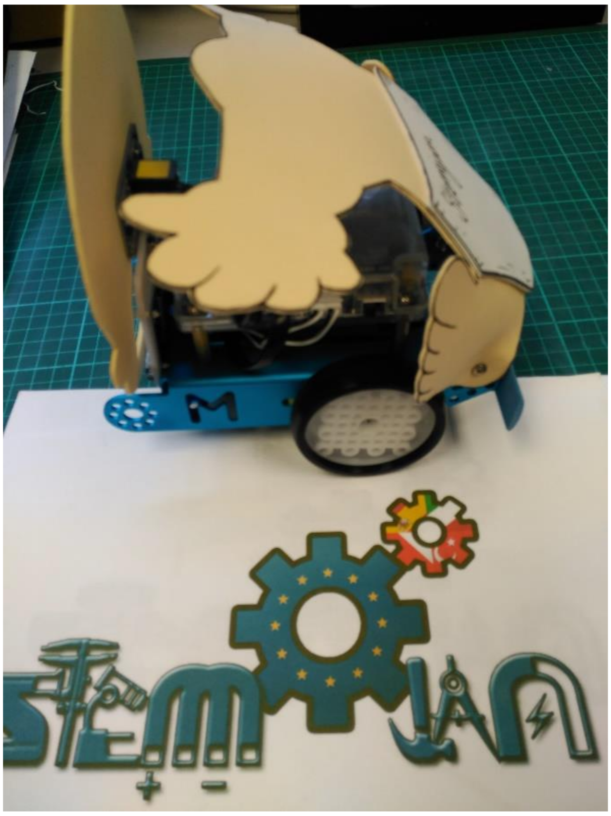


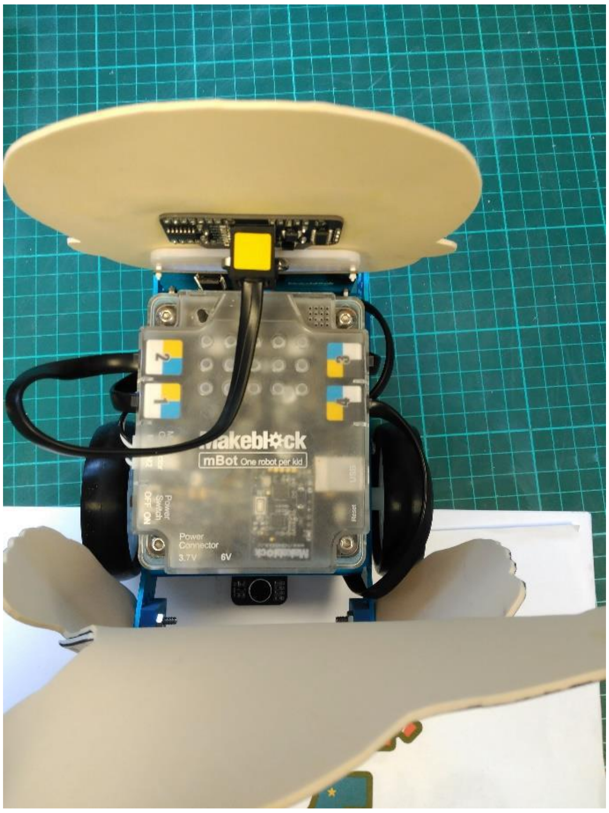
Second version

For the development of the activity, we will need to adapt a disguise to our mBot, which will represent the animal or thing that we want to play with it.





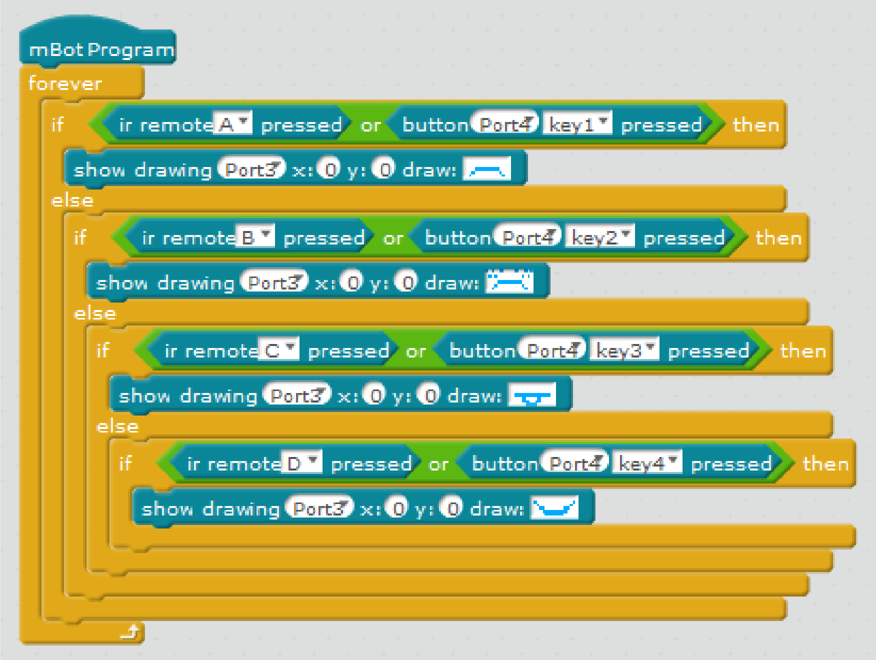




1. Personalize the emotions of the baby:

In the first part, a code has been developed in which the user can personalize the emotions of the baby, since through the remote control or through the Me 4 Button sensor, by pressing a button the baby will show an emotion, and by touching another button, the baby will change the expression again. In this way, for example, small theaters can be made to call attention to the smallest people.

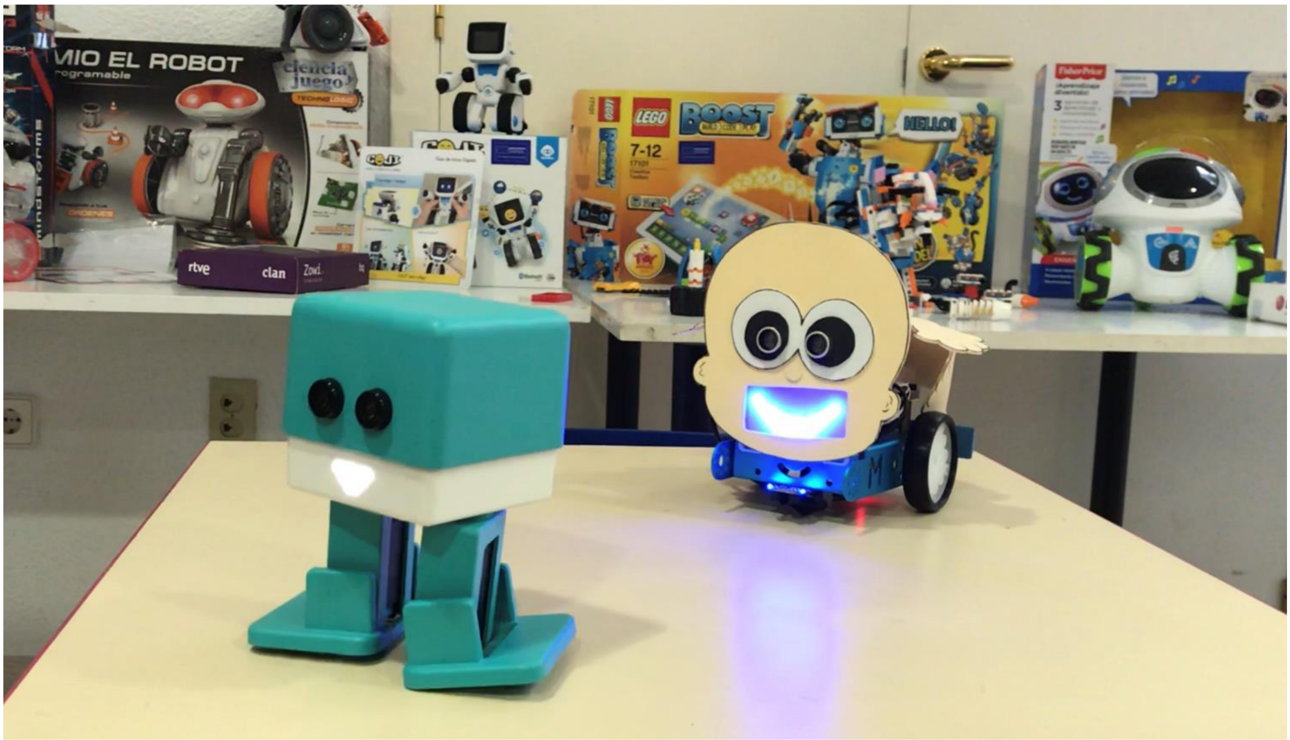
Once we have our atrezzo, we start the programming to the mBot:

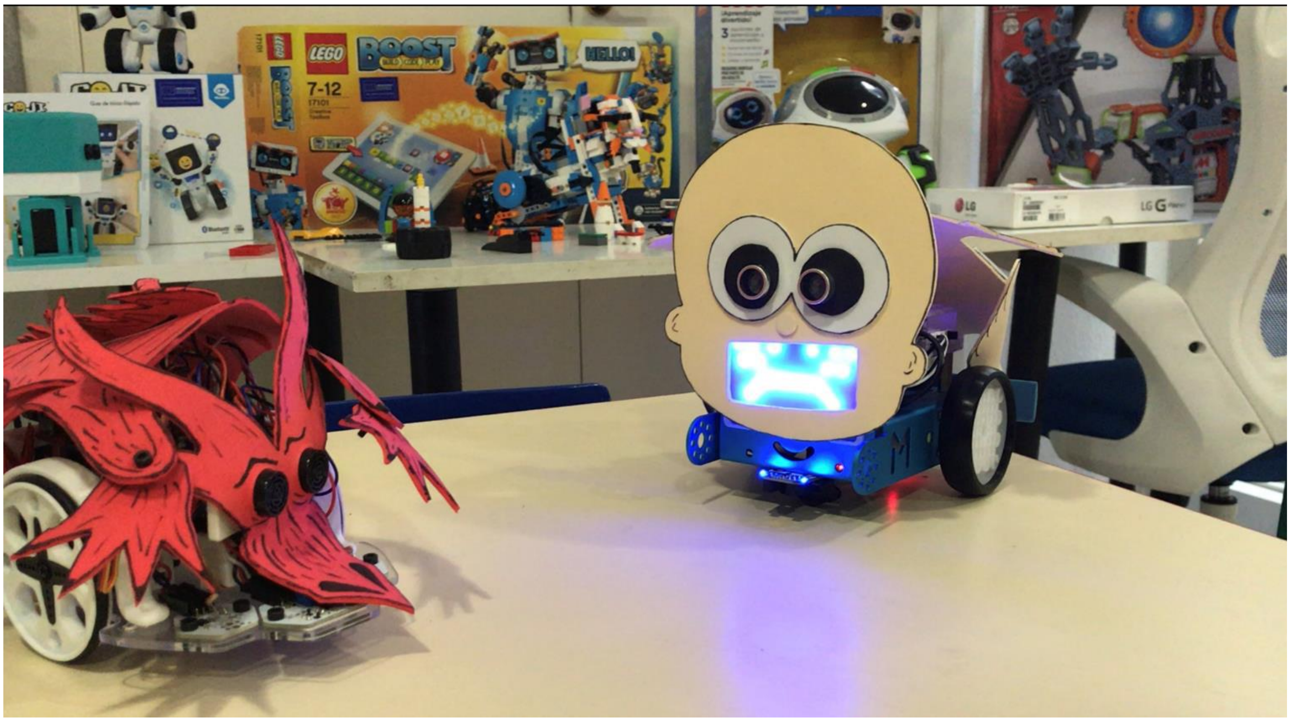


```
mBot Program
forever
  if ir remote A pressed or button Port4 key1 pressed then
    show drawing Port3 x: 0 y: 0 draw:
  else
    if ir remote B pressed or button Port4 key2 pressed then
      show drawing Port3 x: 0 y: 0 draw:
    else
      if ir remote C pressed or button Port4 key3 pressed then
        show drawing Port3 x: 0 y: 0 draw:
      else
        if ir remote D pressed or button Port4 key4 pressed then
          show drawing Port3 x: 0 y: 0 draw:
```

The code is very simple, where it always repeat the loop, and when the user press the remote control button or one of the buttons of the “Me 4 Button Actuator”, the LED Matrix will show the expression that has been determined.

To explain this first section better, we have used different robots, which will appear in front of the Baby STEMJAM and this will show their feelings, which will be chosen by the user through the remote control.





2. Do not wake up the baby:

The second section is that the baby will recognize 3 states, when someone or something approaches (ultrasonic sensor), if there is too much light (light sensor) and there is too much sound (sound sensor). When the baby recognizes this action, it will emit both an acoustic and luminous signal.

The objective is to convert mBot into a real baby, which will adapt to any circumstance, as we will see in the program code.

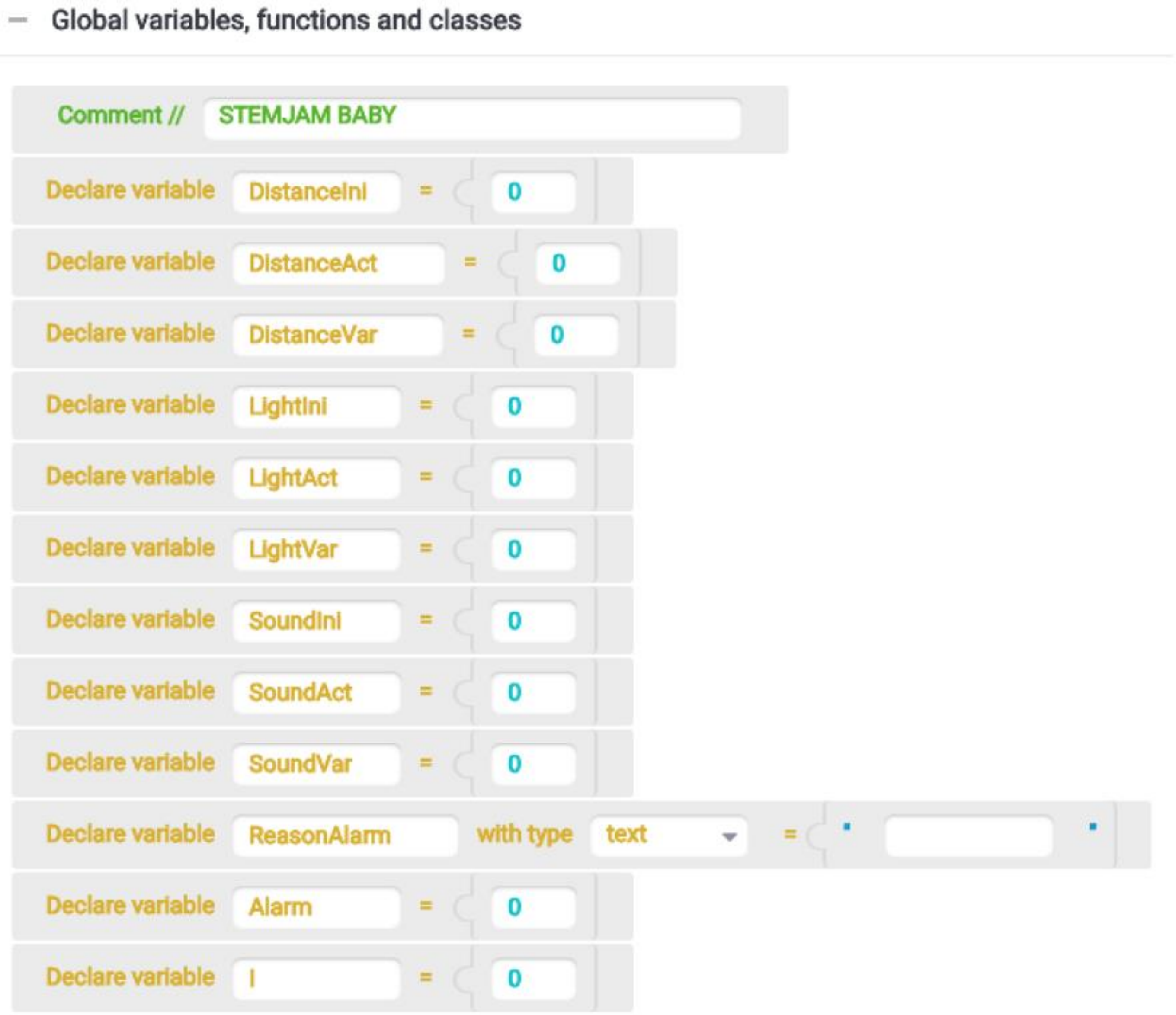
In case of any anomaly, the mBot will emit an alarm and will show in the LEDs Matrix the cause for which it was issued. Once activated, the alarm can be desactivated by pressing the button that is integrated in the board.

To achieve the objective, we will use ultrasonic sensor,the light sensor and the sound sensor.

Now, we detailed the Software:

The code was developed in BitBlock Software (http://bitbloq.bq.com)

1. <u>The variables:</u>



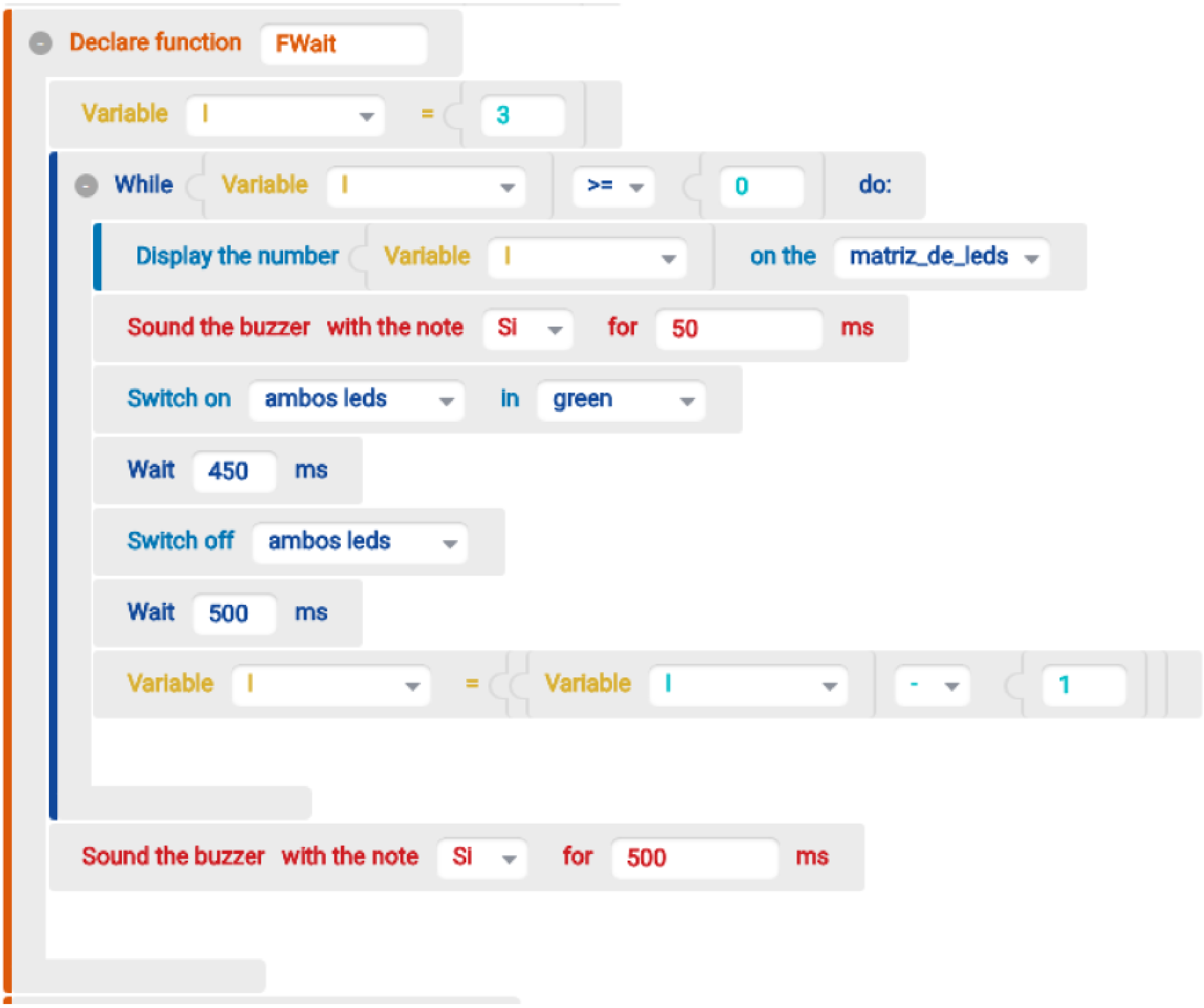
```
— Global variables, functions and classes

Comment // STEMJAM BABY
Declare variable DistanceIni = 0
Declare variable DistanceAct = 0
Declare variable DistanceVar = 0
Declare variable LightIni = 0
Declare variable LightAct = 0
Declare variable LightVar = 0
Declare variable SoundIni = 0
Declare variable SoundAct = 0
Declare variable SoundVar = 0
Declare variable ReasonAlarm with type text = " "
Declare variable Alarm = 0
Declare variable I = 0
```

When starting the program, the mBot will take as reference the current values of distance, light and sound, so any variation that exceeds a margin determined by the user in the program, will cause the alarm to be triggered, so that it will give a greater realism to the activity, since our mBot will be able to adapt to any circumstance.

The variables that we see in the image are the initial value of the distance, the current value of the distance, and the variation of it, which will determine if the margin has been exceeded or not. The same for sound and light.

We also have a variable that will show the reason why the alarm has sounded and whether the alarm is active or not.

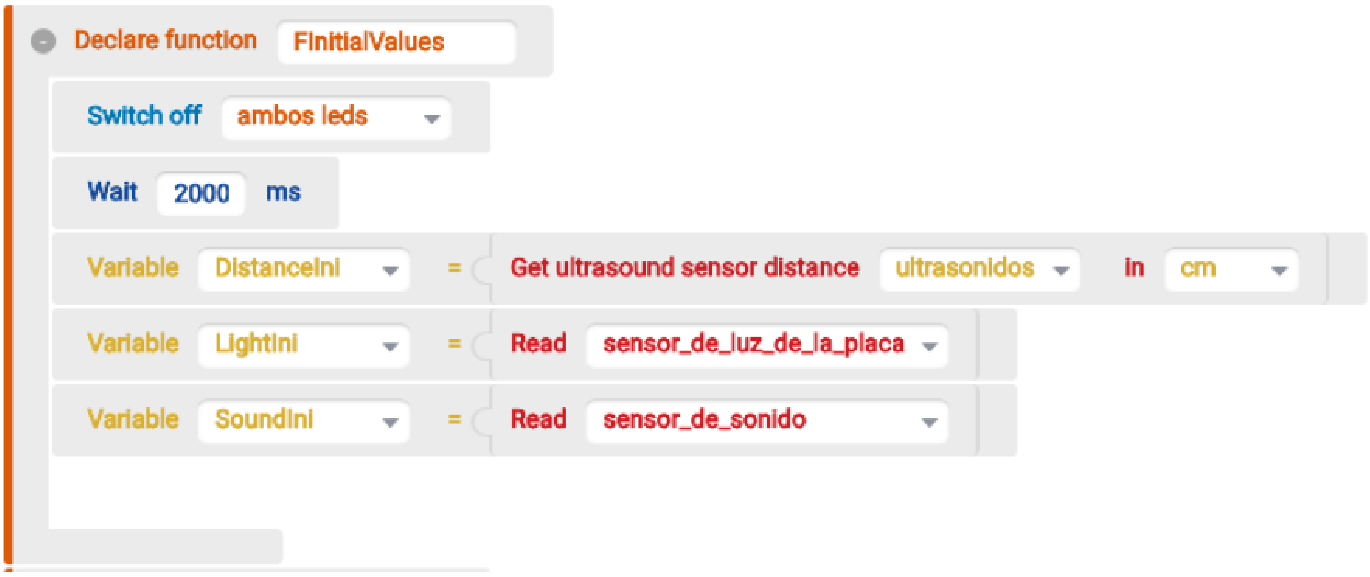
2. <u>Wait Function:</u>



```
Declare function FWait
    Variable I = 3
    While Variable I >= 0 do:
        Display the number Variable I on the matriz_de_leds
        Sound the buzzer with the note Si for 50 ms
        Switch on ambos leds in green
        Wait 450 ms
        Switch off ambos leds
        Wait 500 ms
        Variable I = Variable I - 1
    Sound the buzzer with the note Si for 500 ms
```

The wait function counts down. At the moment when the button on the shield is pressed, this countdown of 3 seconds begins to count, and by the time it reaches 0, the baby will already be exposed to any variation of the environment.

3. Initial Values Function:

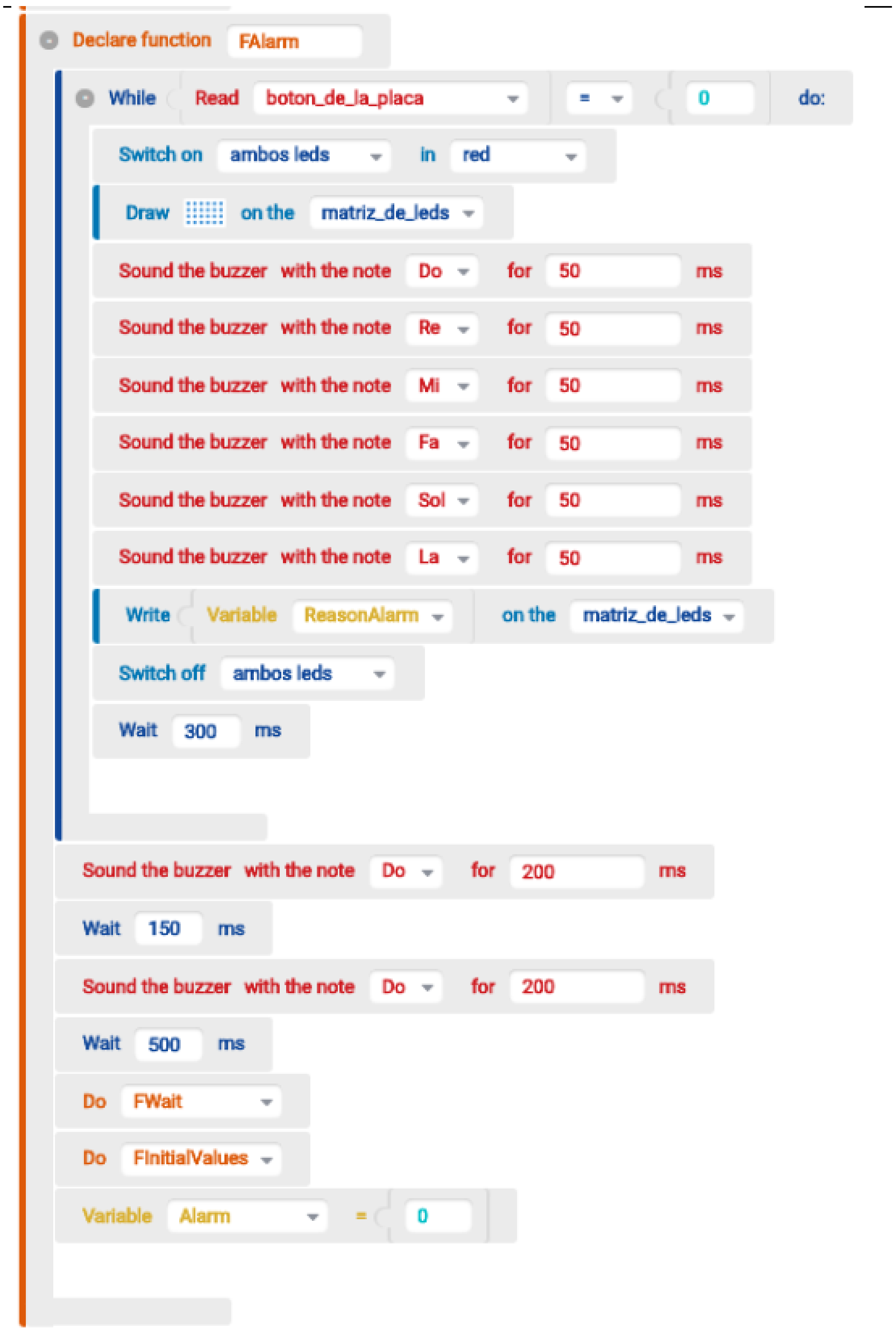


```
Declare function FinitialValues
    Switch off ambos leds
    Wait 2000 ms
    Variable DistanceIni = Get ultrasound sensor distance ultrasonidos in cm
    Variable LightIni = Read sensor_de_luz_de_la_placa
    Variable SoundIni = Read sensor_de_sonido
```

This function establishes the initial values of each sensor.

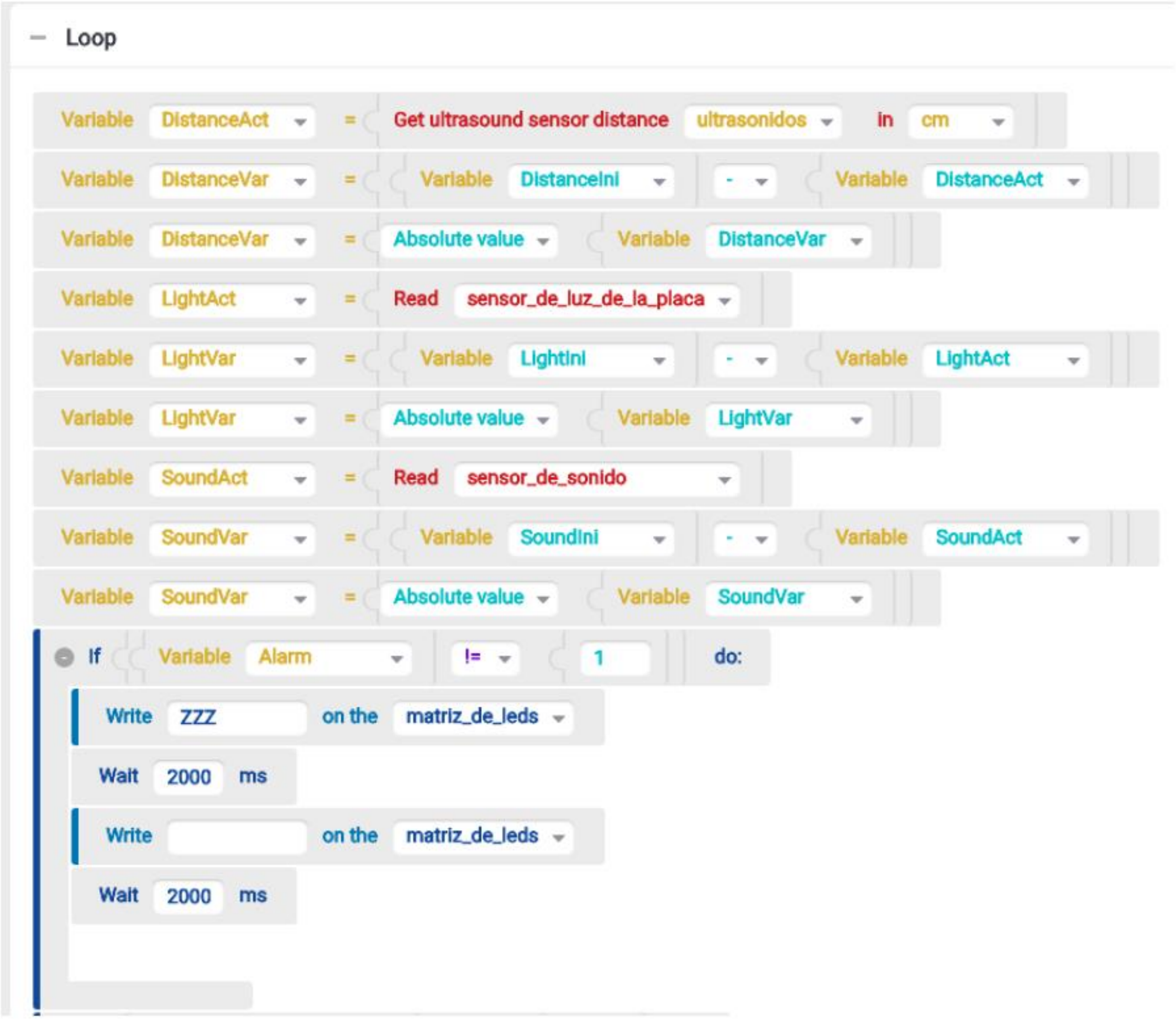
4. <u>Alarm Function:</u>

It is the function that will emit the sound of the alarm and that will stop when the button on the plate is pressed.



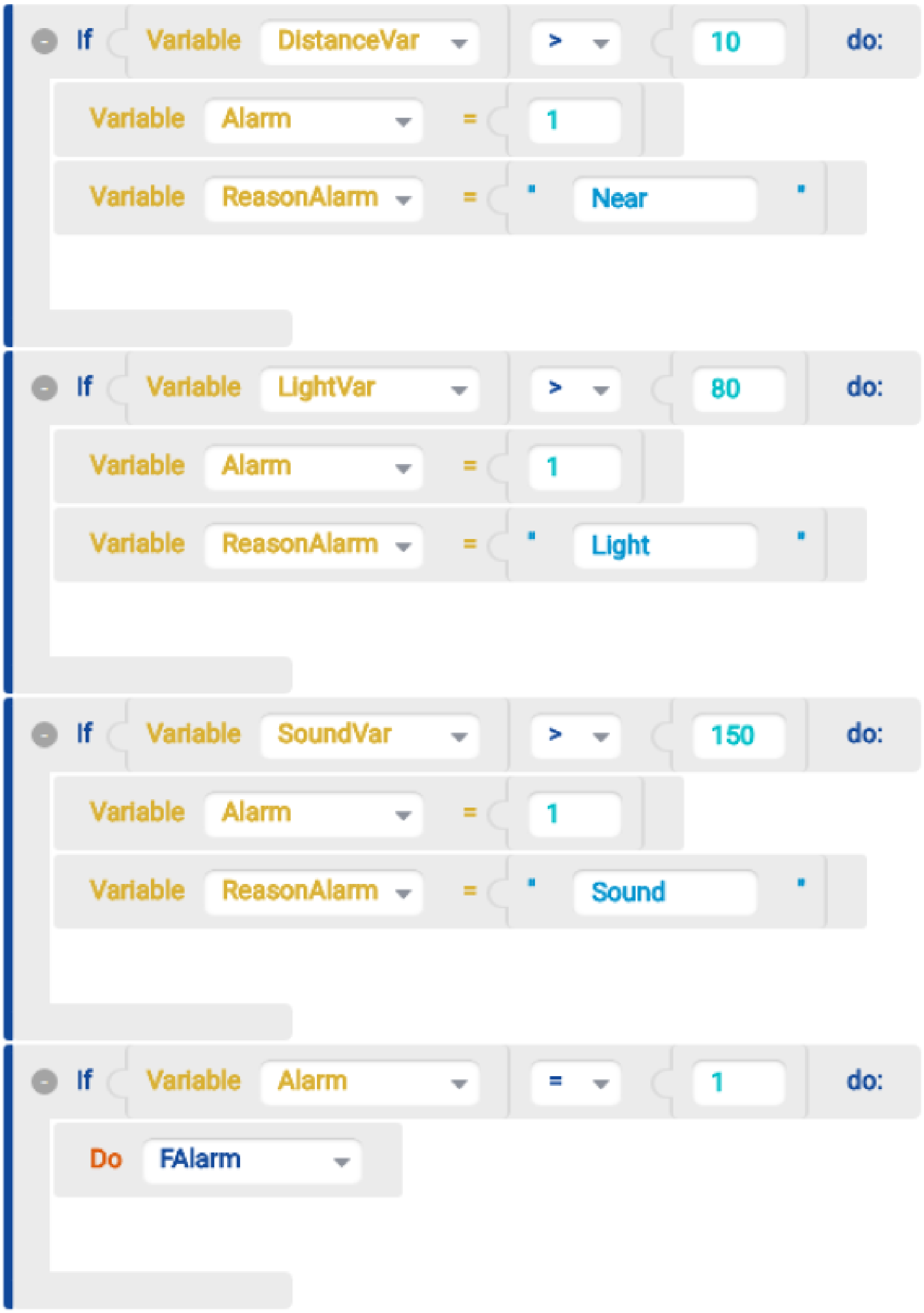
```
Declare function FAlarm
    While Read boton_de_la_placa = 0 do:
        Switch on ambos leds in red
        Draw [::::] on the matriz_de_leds
        Sound the buzzer with the note Do for 50 ms
        Sound the buzzer with the note Re for 50 ms
        Sound the buzzer with the note Mi for 50 ms
        Sound the buzzer with the note Fa for 50 ms
        Sound the buzzer with the note Sol for 50 ms
        Sound the buzzer with the note La for 50 ms
        Write Variable ReasonAlarm on the matriz_de_leds
        Switch off ambos leds
        Wait 300 ms
    Sound the buzzer with the note Do for 200 ms
    Wait 150 ms
    Sound the buzzer with the note Do for 200 ms
    Wait 500 ms
    Do FWait
    Do FinitialValues
    Variable Alarm = 0
```

5. Loop:



```
— Loop
Variable DistanceAct = Get ultrasound sensor distance ultrasonidos in cm
Variable DistanceVar = Variable DistanceIni - Variable DistanceAct
Variable DistanceVar = Absolute value Variable DistanceVar
Variable LightAct = Read sensor_de_luz_de_la_placa
Variable LightVar = Variable LightIni - Variable LightAct
Variable LightVar = Absolute value Variable LightVar
Variable SoundAct = Read sensor_de_sonido
Variable SoundVar = Variable SoundIni - Variable SoundAct
Variable SoundVar = Absolute value Variable SoundVar
If Variable Alarm != 1 do:
    Write ZZZ on the matriz_de_leds
    Wait 2000 ms
    Write  on the matriz_de_leds
    Wait 2000 ms
```

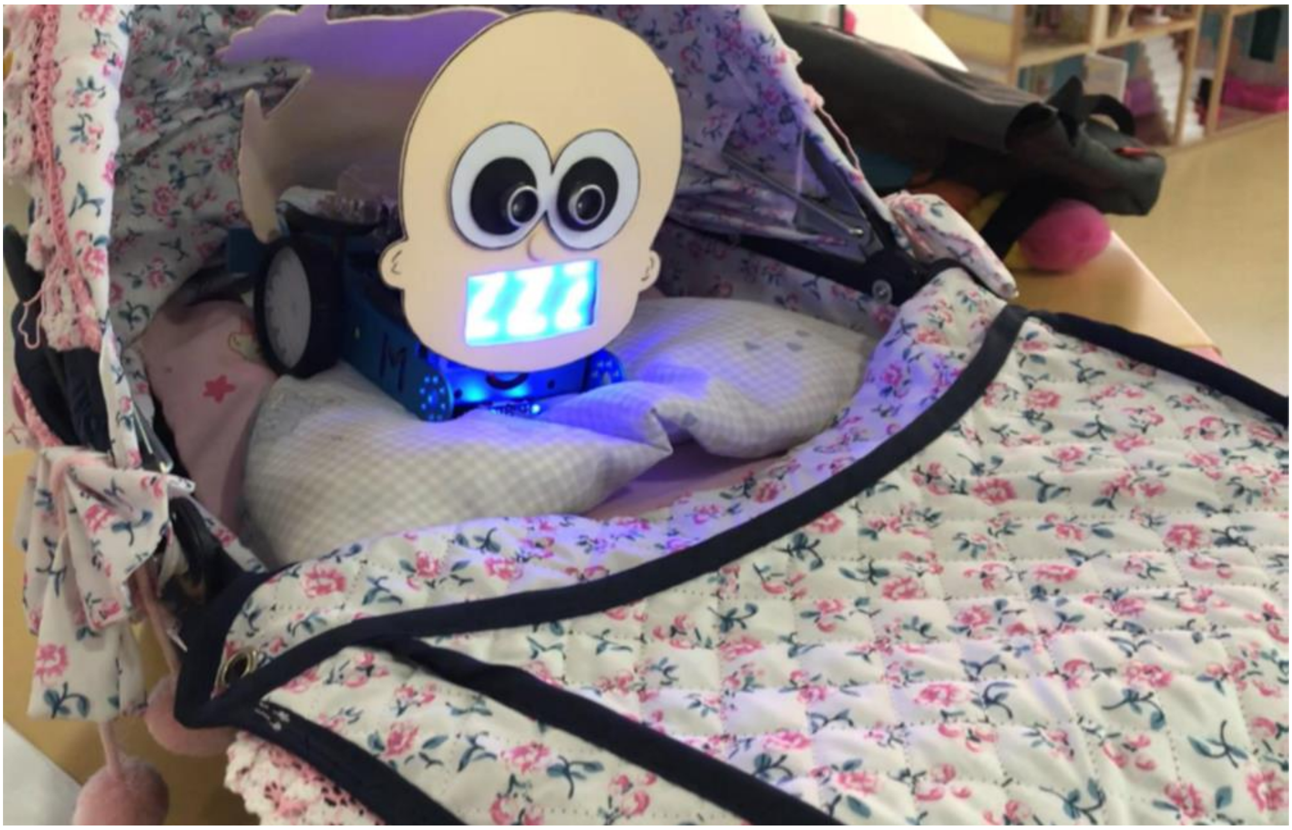
In this part of the code, which is executed at the moment, the “DistanceAct” variable is queried if the current distance has changed a lot from the “DistanceIni”, so if there is too much variation, the alarm will be activated as we will see in the following image. (For sound and light it is the same process).



```
If Variable DistanceVar > 10 do:
    Variable Alarm = 1
    Variable ReasonAlarm = " Near "
If Variable LightVar > 80 do:
    Variable Alarm = 1
    Variable ReasonAlarm = " Light "
If Variable SoundVar > 150 do:
    Variable Alarm = 1
    Variable ReasonAlarm = " Sound "
If Variable Alarm = 1 do:
    Do FAlarm
```

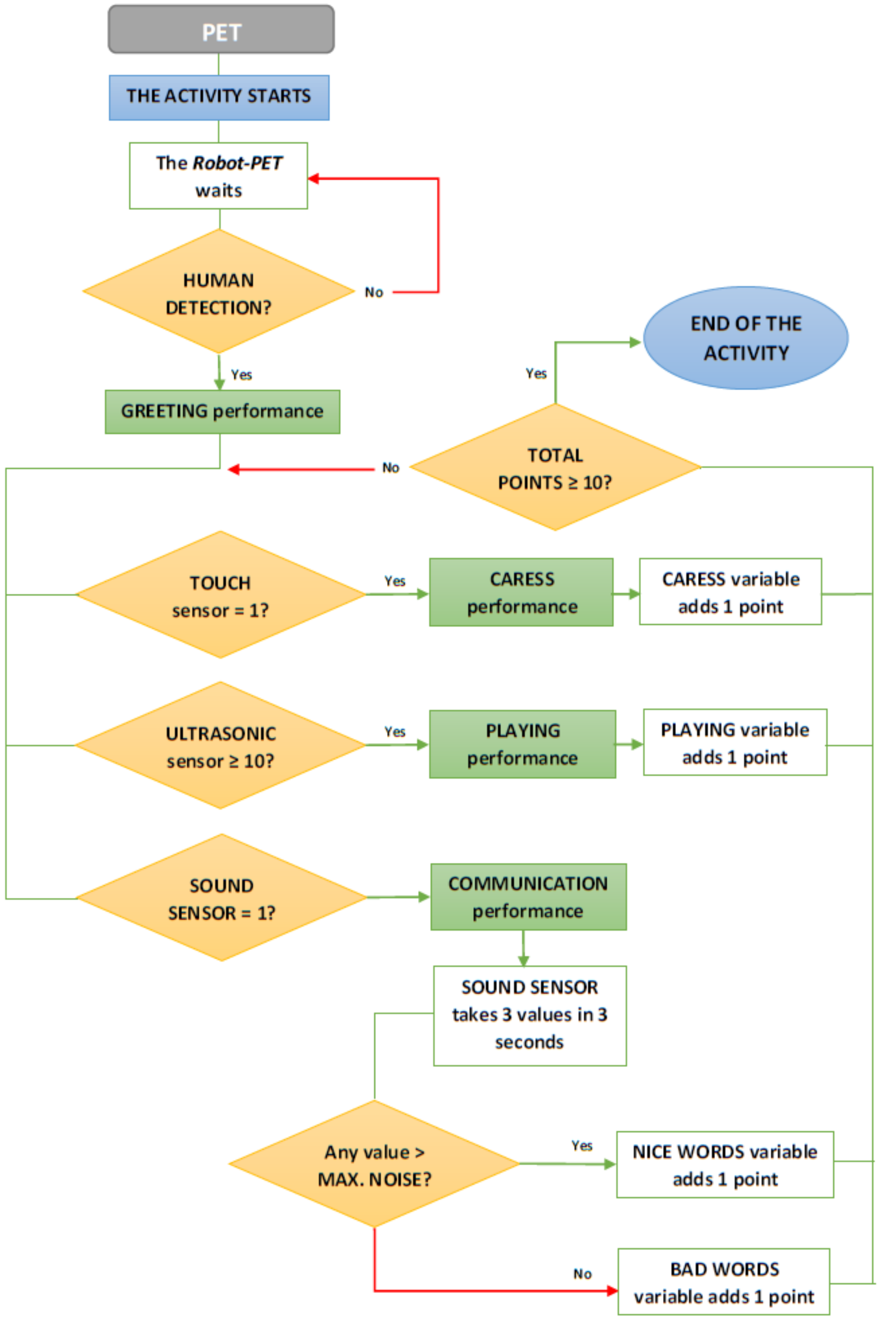
When the variables "xxxVar" are greater than the value predetermined by the programmer, the alarm will be activated and the "ReasonAlarm" variable will show because it has been executed.

Now, we will show some pictures of the activity:

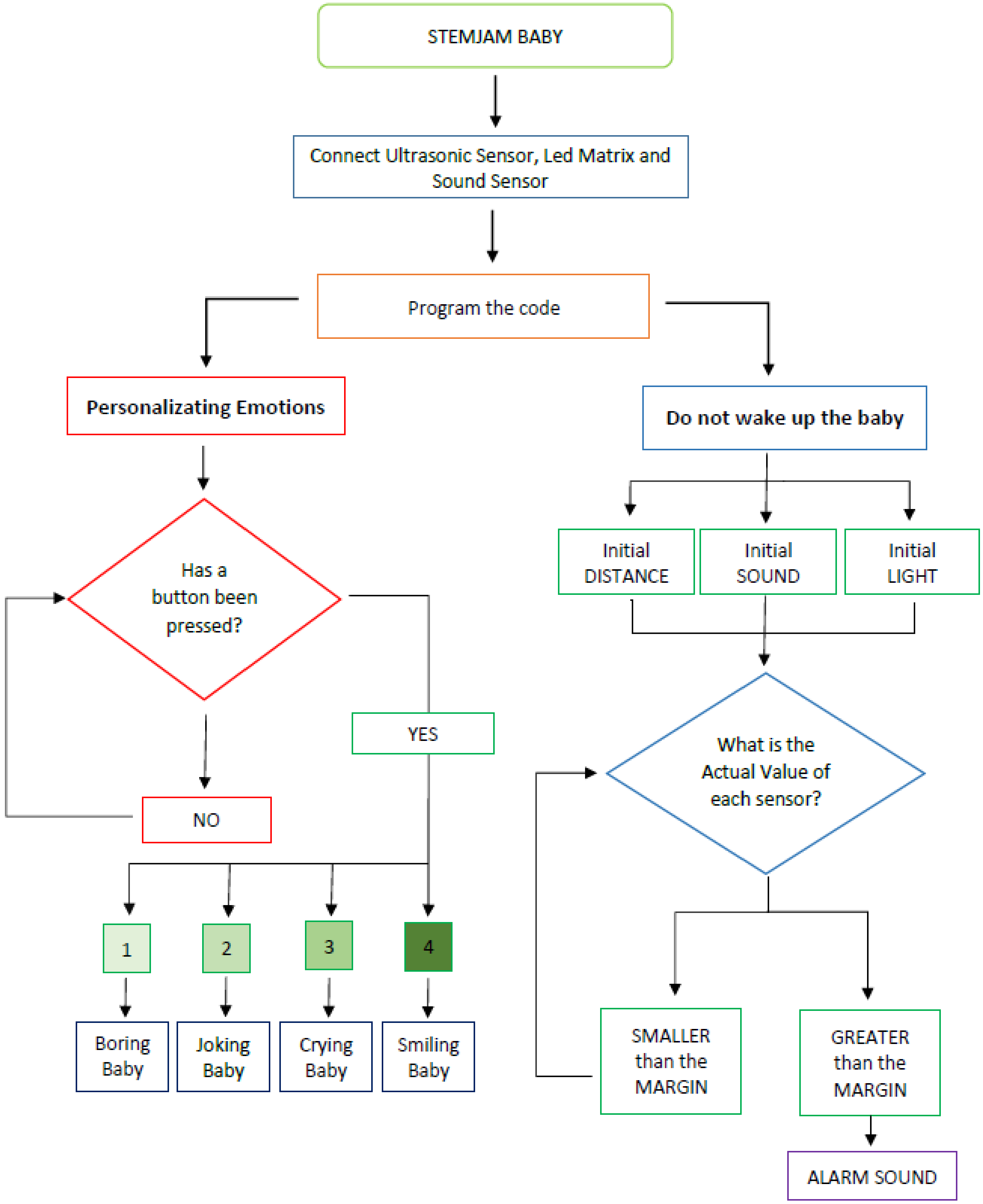


FLOW CHART

First version



Second version



STUDENTS' EVALUATION

For the evaluation of the students in this activity, use the Evaluation Rubric designed for this project.

BIBLIOGRAPHY

"Jugando con MBlock". Makeblock España

"Divirtiéndome con MBot". Susana Oubiña

"Prácticas mBot". Javier Fernández Panadero

Comunidad de Makeblock en español. (http://www.makeblock.es/foro/)

SCALABILITY

The scalability of this activity can be as difficult as you want, always based on making the mBot as real as possible to the being that we imitate.

MORE INFORMATION

DIFFICULTIES:

- ❖ DETERMINATING HUMAN VOICE TONE: it was very difficult to establish a pattern in order to differentiate between nice and unpleasant talking. That's why after reading medical papers about voice modulation and music articles about tones it was decided that two variables were involved (time and level sound), and by combining them it was possible to approximate this value collection.

- ❖ PROGRAMMING TOUCH SENSOR: out of lack of awareness about touch sensor functioning, it was hard to make the sensor work. The led it incorporates gave signs of activity, but the sensor just could not work if it was not equalized to a binary value ("1").