

What is WeGoSTEM?

Are you an elementary school teacher teaching ages 10 to 12? Do you want to enthuse students about technology and computer science? Then you have come to the right place at WeGoSTEM. We offer all the materials you need to build and program a robot with your students. A robot that can draw. Your students will acquire a whole host of STEM skills, from engineering to computational thinking.

That sounds almost too good to be true. Yet it works. More than 25,000 children all over Europe have already worked with WeGoSTEM. The curriculum is structured, fits well with the capacities of the students, and fits exactly into two one-hour class periods:

First hour

- a class-based discussion about robots (10 - 15 min);
- programming a human being (10 - 15 min);
- building a drawing robot (25 min);

Second hour

- programming the drawing robot and experimenting (40min);
- clean up (10 min).

How does it work?

Are you eager to get started? Then read on. In the next sections, we will walk you through the WeGoSTEM learning pathways. You'll discover the various activities we've developed for your students. Each activity is accompanied by a brief explanation which prepares you to support your students.

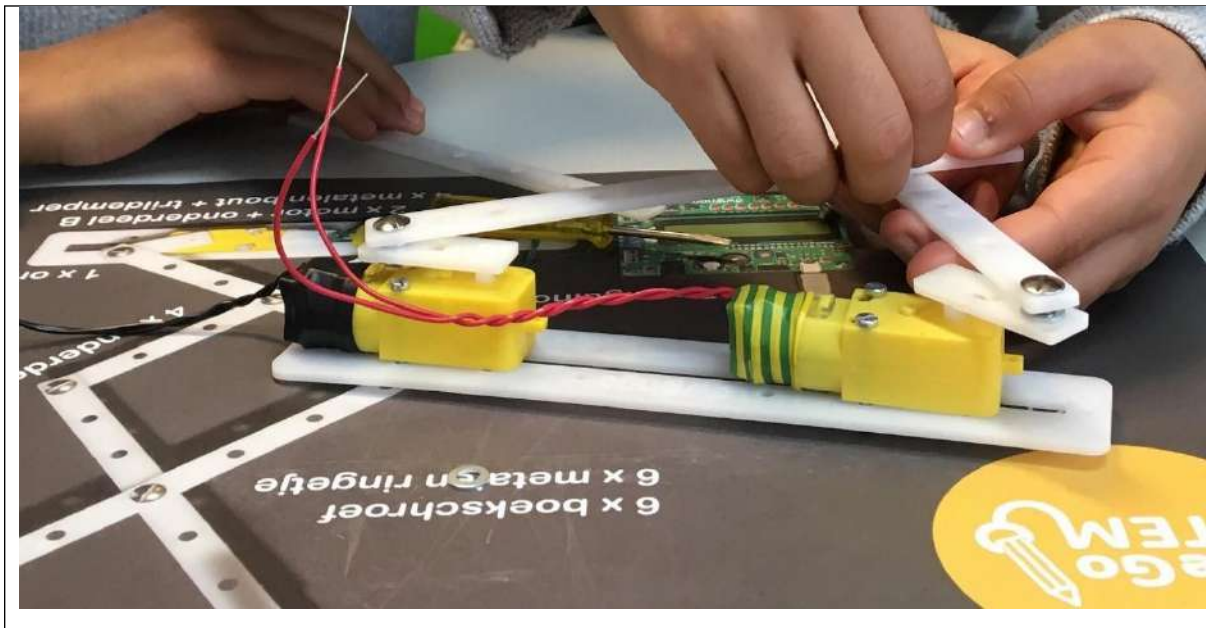
If you have additional knowledge about robots, feel free to share it with the students! The time indication is indicative, and can of course be adapted to the needs of the students.

In any case, children are blazingly enthusiastic when they can go home with their own robot drawing!

If there are any questions after the learning path, you can always contact us at

scholen@dwengo.org.

Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them. Project Number 2022-1-CY01-KA210-SCH-000084731



What will you be doing?

You are going to be challenged to build and program your own drawing robot!

Before you get started though, you will have to complete some super cool assignments to give you a better idea of how to go about it.

These assignments include explanations about robots, activities such as "Programming a Human Being" and programming exercises.

Have fun!

Preparation- What is a robot?

10 - 15 minutes in class

What happens in class?

This section briefly provides some context regarding the history and construction of robots. This builds students' understanding of the usefulness, construction and operation of robots.

Materials

Photos of different robots [can be downloaded here](#).

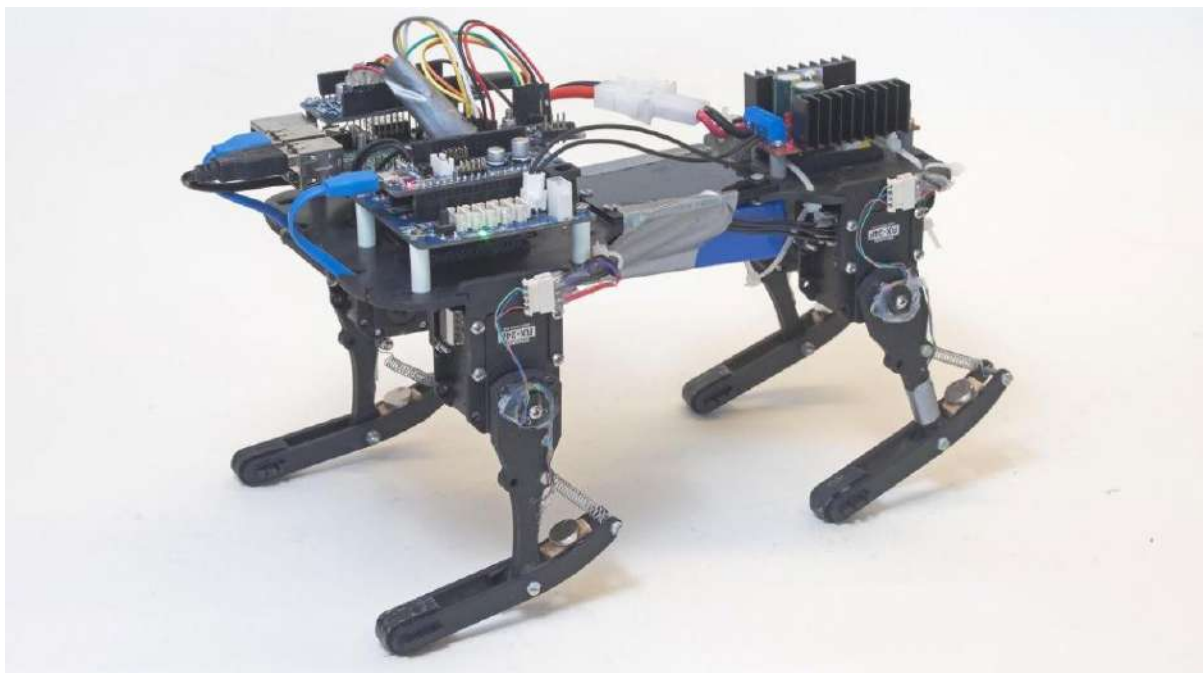
What is a robot? What robots have you already seen?

In 1495, Leonardo da Vinci designed a "mechanical lion" that could propel itself using clockwork mechanics. However, it wasn't until 1961 that the industry welcomed the first robot: the Unimate.

This robotic arm could be programmed to perform tasks independently. Advanced variants of the Unimate are still used today, for example in car manufacturing or microsurgery.



In parallel with the development of such industrial robots, researchers started building robots with legs in 1950. Some famous descendants of these are the Atlas humanoid by Boston Dynamics (which has a human form) and Sony's Aibo (a robot dog).

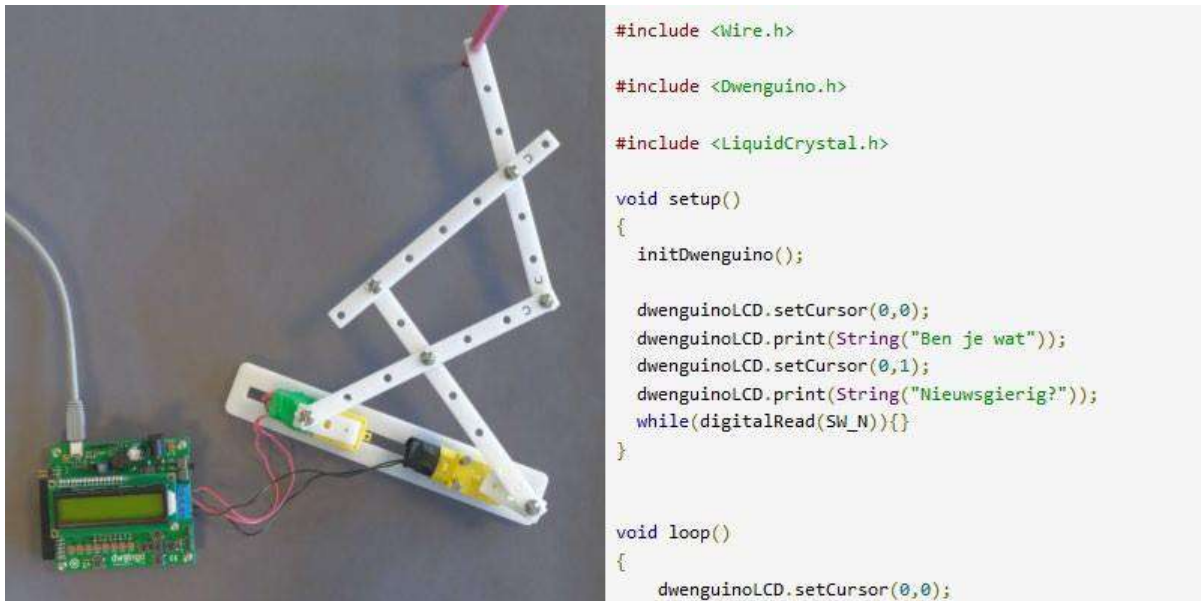


Moreover, researchers are not only concerned with mechanical construction: they are also increasingly focusing on the social skills of robots. KISMET from MIT, from the 1990s, for example, smiles at you. Robots with facial expression are used as front desk clerks in hotels or hospitals, among other things.



What does a robot consist of?

A robot consists of hardware (its body, with wiring, computing unit and batteries) and software (the program that controls it).

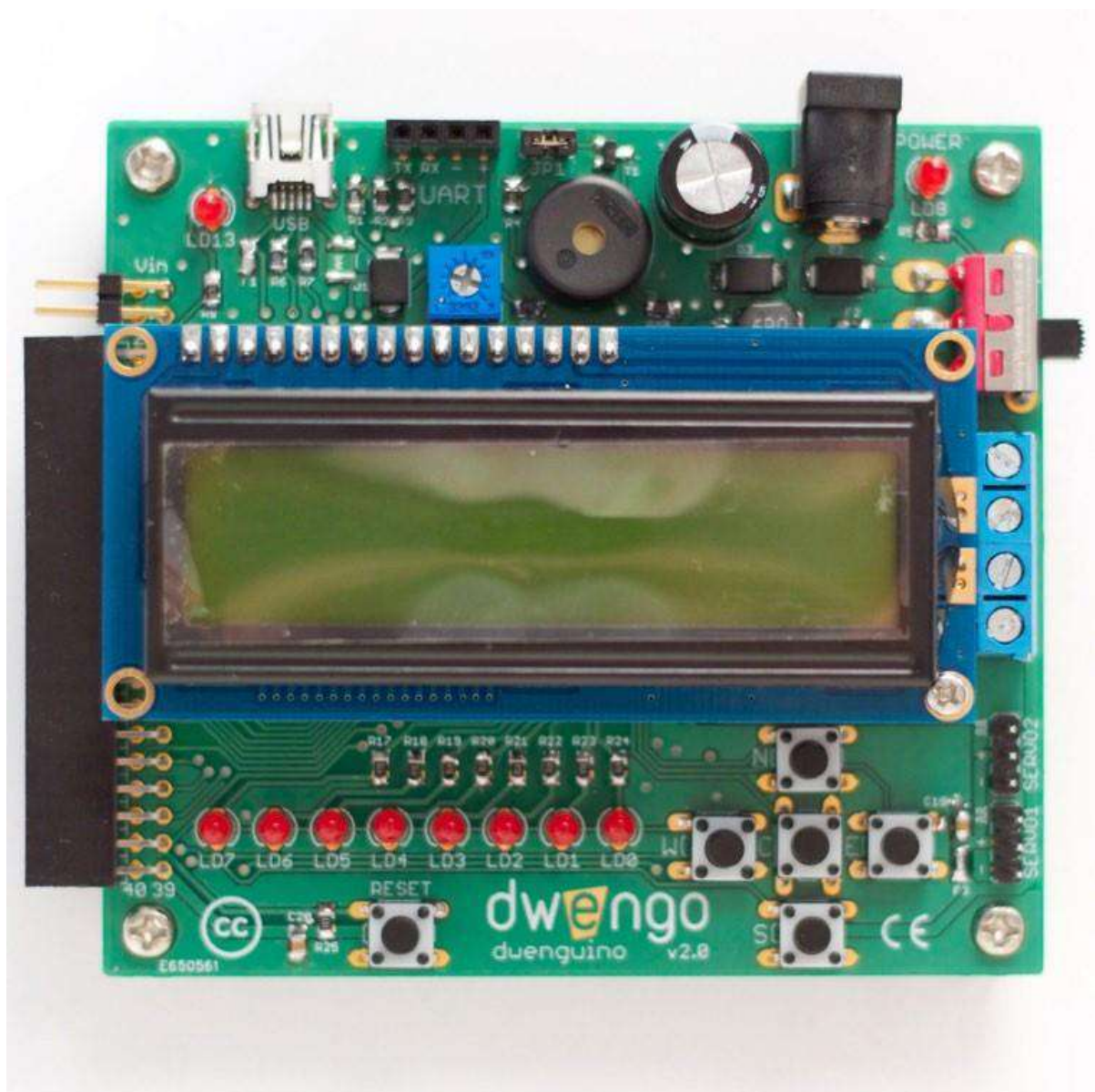


The 'body' of a robot contains mechanical parts that can be made of many materials: wheels, an arm, a head, ...

The robot has sensors (inputs) to "feel" and actuators (outputs) to "act". One can equip a robot with distance sensors, ground sensors, touch sensors, light sensors or sound sensors. Examples of actuators include an LCD screen, a buzzer or a servo motor.

Your computer also has inputs and outputs. The keyboard and mouse are examples of inputs; the screen is an output.

The computational unit is the brain of the robot, it is used by your robot to make "decisions. For this, you use a processor (like in a computer) or a microcontroller (like on the dwenguino), for example.



The wiring ensures that the computing unit, sensors and actuators are connected. How the robot controls its actuators depends on the information it collects from its sensors and how the computing unit is programmed. The batteries in the robot supply the motors with the necessary power.

A robot can perform a certain task only if its computing unit is programmed to do so. What the robot does is not necessarily fixed: you can reprogram the computing unit so that the robot performs a different task. Moreover, that task may depend on the information that the microcontroller receives through its inputs.

Wondering what your drawing robot is capable of doing? Let's find out!

Robot versus animal?

A robot is sometimes compared to an animal. That comparison leads to interesting questions. Does an animal have sensors? What part of an animal can you compare a robot's motors to? And does the wiring have an animal equivalent?

This is a matter for discussion, but here is a possible way of comparing animals to robots:

- The computing unit is the brain of the robot and corresponds to the brain of an animal. This is because a robot can only perform a task if its computing unit is programmed to do so.
- The motors make the robot move, so you could compare them to an animal's muscles.
- The sensors are the robot's senses. Through its sensors, the robot retrieves information from its environment.
- Part of the wiring forms the robot's nervous system. Through these wires, signals are sent: on the one hand, from the computing unit to the motors and sensors, and on the other hand, from the sensors to the computing unit.
- Another part of the wiring takes care of transporting the necessary power. These are like the blood vessels of animals. Animals get energy from their food, this energy for example is needed to move their muscles. Robots get their energy from their batteries. They use the electricity provided by the batteries to move their motors.

So how does an animal differ from a robot? The difference lies mainly in the way an animal makes decisions. An animal, consciously or unconsciously, is much more flexible than a robot. A robot moves only when its computing unit commands it to do so. If something is wrong with the program, if a sensor is broken or if the robot has too little energy, the robot does not do what it should. This is not true for an animal, since the animal brain is capable of coming up with solutions to problems.

Preparation- Programming a human being

10 - 15 minutes during class

What happens in class?

Students program each other to make a drawing. One student is the 'programmer' and gives careful instructions to the other students to reproduce a particular drawing. The other students are "the computer". How quickly and accurately is the drawing completed?

Why this exercise?

Students learn how to solve a relatively difficult problem by breaking it down into small steps. They must give very precise instructions and do so in the correct order. Focus: Keep an eye on time!

Preparation

Materials

Pen and paper.

Pictures for the exercise [can be downloaded here](#). Start with the simplest drawing!

Programming a human being



Introduction

For robots to do meaningful things, they must be programmed. That is, they must be given unambiguous instructions. After all, robots cannot interpret and literally carry out any instruction you give them. When writing instructions is called “programming”.

The programmer's challenge is to solve problems by breaking them down into small steps that are executable by the computer.

Materials

For this assignment, you need only pencil and paper.

Skills

A robot only does what you program. So you have to give very precise instructions. You learn to solve a relatively difficult problem by breaking it down into small steps.

Assignment

Choose one programmer. The others are robots. The programmer describes the drawing to the class and the robots draw it. The robots are not allowed to talk or ask questions during the drawing process!

The goal is to see how quickly and accurately you can make the drawing. So do not show the original drawing to the robots until they have finished drawing!

1. Start by using a simple picture for this exercise.
2. Discuss the result. What went well? What went less well? Are all the figures the same size? Were you able to perform the instructions in different ways?
3. Repeat steps 1 and 2 with other programmers and images.

Conclusion

You probably noticed that it is not always easy to give clear instructions. For example, if you want a specific house to be drawn by a computer, "Draw a house" is much too vague for the computer. What is the shape of the house? How big should the house be? Where on the sheet should the computer start drawing? You found that out for yourself during the assignment. It may have been frustrating for the robots that they were not allowed to ask questions. But of course, a real robot can't do that either!

Computers can't improvise. They carry out every instruction you give them verbatim. So as a programmer, you have to break down commands into small, clear steps. Steps that the computer can execute. And, of course, you have to relay the instructions to the computer as clearly as possible.

Preparation- Building a drawing robot

20 - 30 minutes during class

What happens in class?

In this activity, students get to work with physical equipment for the first time. They learn about the different components of a robotic arm and have to assemble them correctly. While doing so they build their own drawing robot.

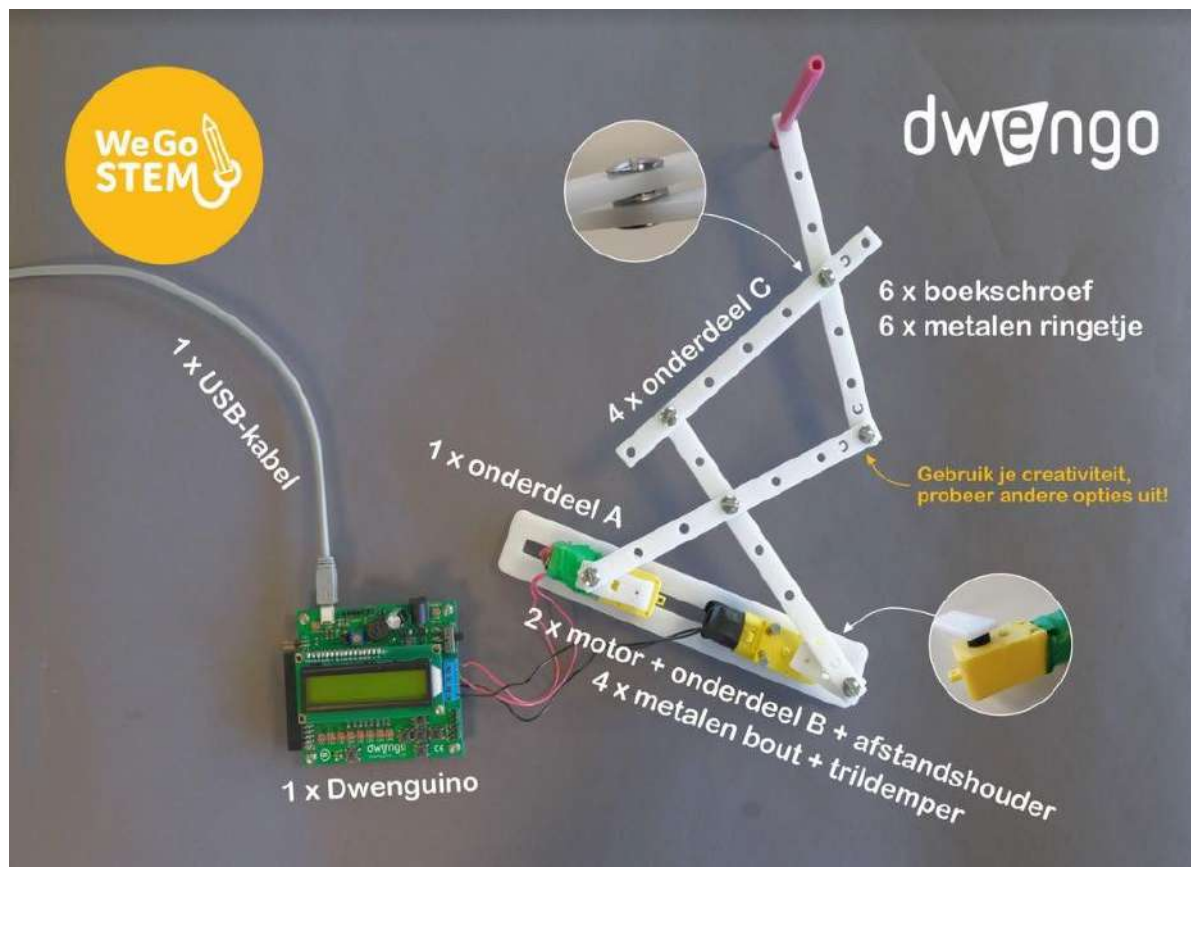
Each group is given a construction diagram. Based on this diagram, the children can usually figure out for themselves how to build the drawing robot. If they do get stuck, you can provide some additional explanation. The steps needed to build the robot are shown in [the following video \(in Dutch only\)](#).

Preparation

Materials for the classroom

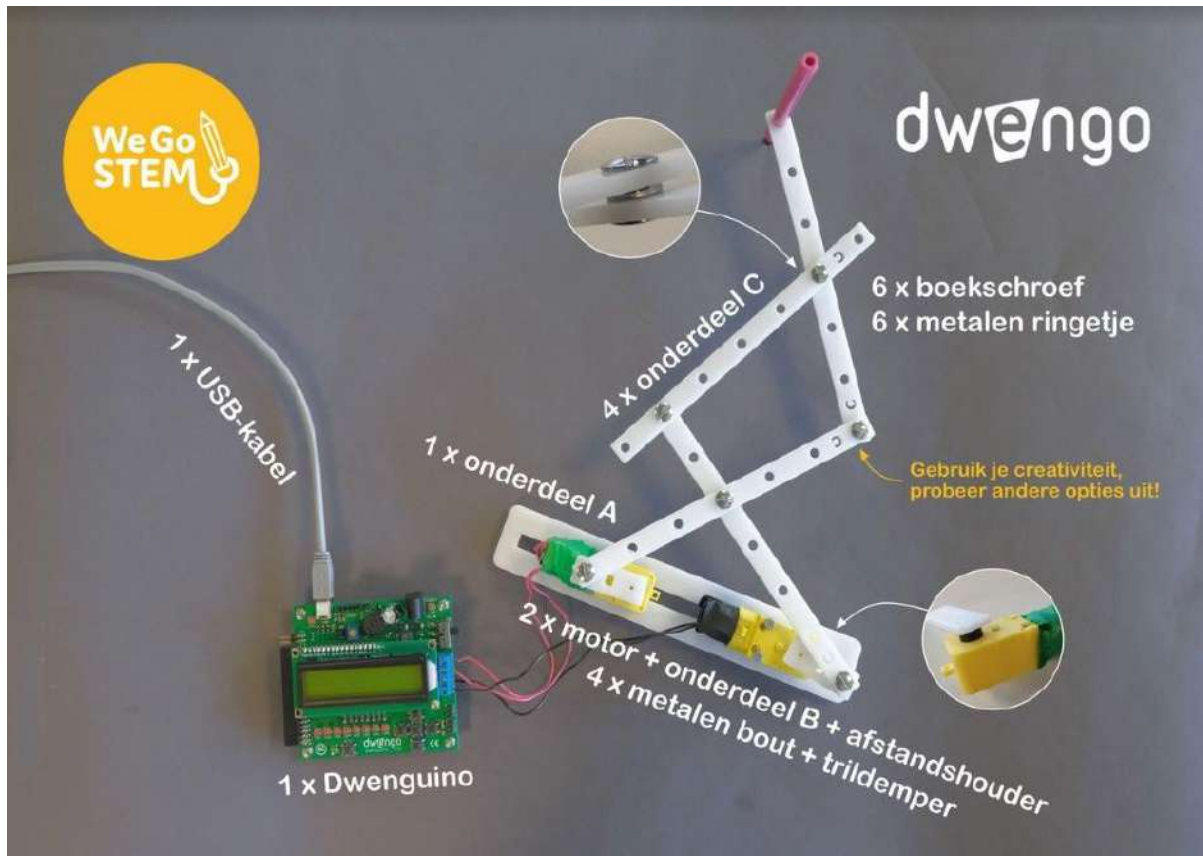
Box of robot components

Photo of a finished robot



Building a Drawing Robot

Now that you know how a robot works, you can start building the robot itself! To do this, use the picture and the parts from the box.



Will you manage to build the robot just by looking at the picture? Do you need some extra help? Then watch [this video \(in Dutch\)](#).

Preparation- Programming the Drawing Robot

40 min during class

What happens in class?

The students now have a robot and want to activate it. But first they need to program it. This is the subject of the next steps of this learning pathway.

Step by step

Students will learn to work independently and develop an inquisitive mindset. First in a simulator, then in real life. In a number of steps, they discover different ways to control their robot. For example, they learn:

1. Writing their name on the LCD screen.
2. Making the robot start only when a button is pressed.
3. Running the motors so the robot draws.
4. Experimenting with the mechanics and speed of the motors.

Tip: Have students program only when they have finished building their drawing robot. As a teacher, you don't have to spoon feed your students. Feel free to let them experiment.

Preparation

It is important that the children get started on their own in the learning path. They learn to work with the LCD screen, the wait function and buttons. For each of these functions, students are given information and exercises. These instructions should suffice.

Should the students still have questions, they can approach the teacher. We therefore recommend that you, as a teacher, are familiar with the basic exercises of the course. This way you will be able to guide the students in the best possible way. You don't have to follow the entire track. Only the parts about the LCD screen, the wait function and buttons are relevant.

Materials for the classroom

Computers with internet connection.

Pre-built drawing robot.

Programming the drawing robot

Now that your robot is assembled, we're going to program it. We're going to tell it very precisely what to do. But because the robot doesn't speak human language, we'll first learn programming language. We'll do that in dwenguinoBlockly, a programming environment for kids like you!

dwenguinoBlockly has two big advantages:

1. It is age-appropriate (third grade primary and first grade secondary).
2. It is a graphical programming language. Programming is as simple as dragging blocks on the screen.
3. There is a simulator. On the screen you can already see what the robot will do.

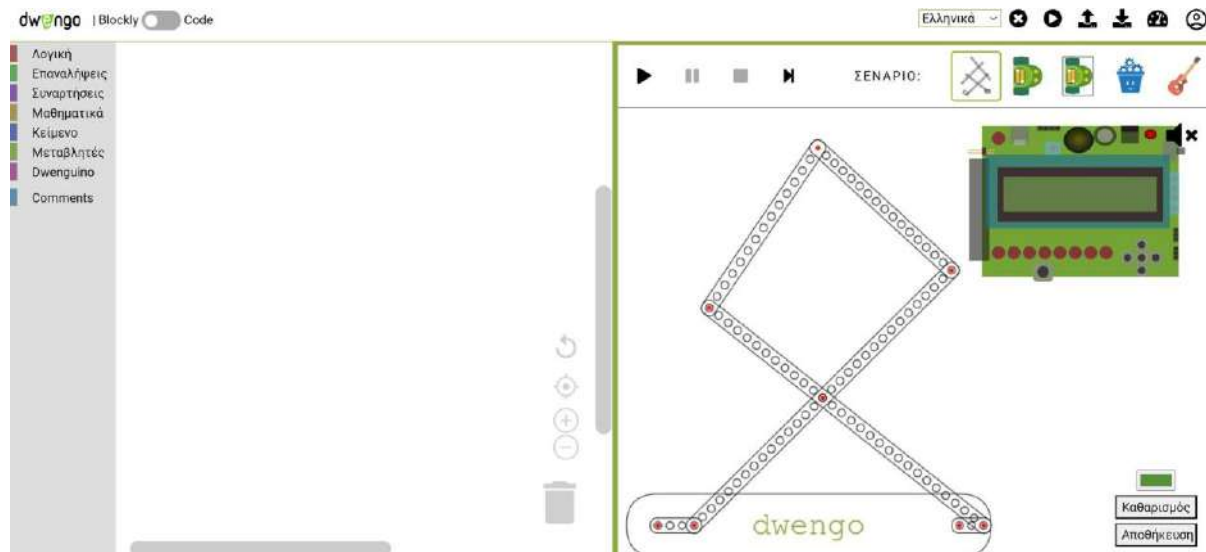
In the next steps of this learning pathway you will learn to work with dwenguinoBlockly. First in the simulator, then with your own robot.

Good luck!

dwenguinoBlockly: A programming environment

The programming environment with simulator is available online at <https://www.dwengo.org/dwenguinoblockly> .




Below is a screenshot of the environment with the description of the various components.





1. The *toolbox*: In this menu you will find the different code blocks. The menu is divided according to categories, each containing a specific kind of blocks. For example, in **Dwenguino** you can find all the blocks to control the drawing robot.
2. The *code* field: Here you can find the program you are creating. The 'get ready/repeat' block is already there.



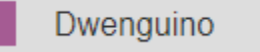
Only code placed in the 'get ready' and 'repeat' sections of this block is executed. Code in any other place will not be executed. So to program, drag and drop blocks from the toolbox into the code field and snap them into the 'get ready/ repeat' block.

3. *Main menu*: This menu allows you to perform actions such as saving your code (using ) , loading it back in (using ) , or opening and closing the simulation environment (using ) .

4. The *simulator menu*: Here you will find the buttons to start and stop the simulation with the buttons  and . It also allows you to choose a specific scenario in which to run your code.
5. The *simulation window*: In this window you see a virtual robot and often a virtual microcontroller board, the dwenguino, on which you can run your code. In the picture, the drawing robot scenario is selected. At the top you see a virtual dwenguino board, at the bottom a virtual drawing robot that you can program.


So in the toolbox you can find the blocks you need to create programs. You have to drag these blocks from here and then place them in the desired order.

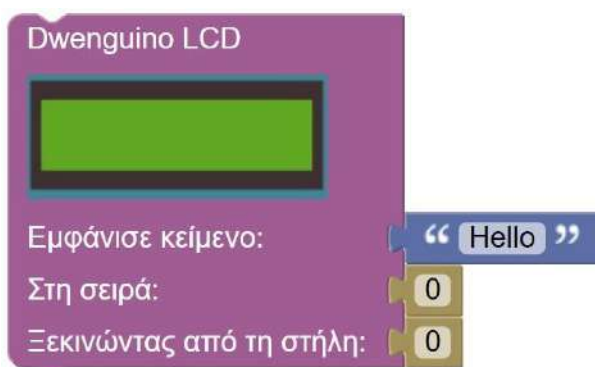
Throughout the exercises you will need new blocks, you find these blocks in the toolbox. The


 Dwenguino toolbox for example will contain blocks that you will need for your robot.

Using the simulator

Now that you know what's where, you can start programming!

- In the simulation environment, choose the drawing robot scenario (spirograph).
- In the  Dwenguino category, look for this block:



- Drag this block into the code field and snap it into the "get ready" section of the "get ready/ repeat" block.
- You just wrote your first program!
- Run this program with the simulator by clicking on the play button  in the simulator menu.

After this exercise you will know the basics. You can take blocks from the toolbox and add them to a program in the code field. You know how to execute that code in the simulator and you see what the code does using the simulator.

Once a program works in the simulator, you can also try it out on a real dwenguino! The following section describes how to upload a program from the simulator to the dwenguino.

Uploading code to the dwenguino

The [video \(in Dutch\)](#) shows you how to upload a program from any computer and browser to your dwenguino. It is important that you go through all the steps as shown in the video!

- Write the program;
- Download the program;
- Open downloads, there is now a .dw-file there
- Connect the dwenguino and press RESET + SOUTH, then release the reset button;
- The dwenguino is between your folders as a USB drive;
- Copy the .dw file to the dwenguino;
- Press RESET

LCD screen

The LCD screen can be used to display text. This can be useful, for example, for reading the values measured by the sensors.

The dwenguino's LCD screen can display up to 32 characters, for example letters or numbers, spread over two lines. So you can display 16 characters per line. The brightness and backlight of the screen are also adjustable, but this will not be discussed here.

Example LCD display

SOLUTION 1.

Show "welcome robot" on the LCD screen.

Solution:



You can customize the text "Welcome robot". The two zeros mean: first line, first character.

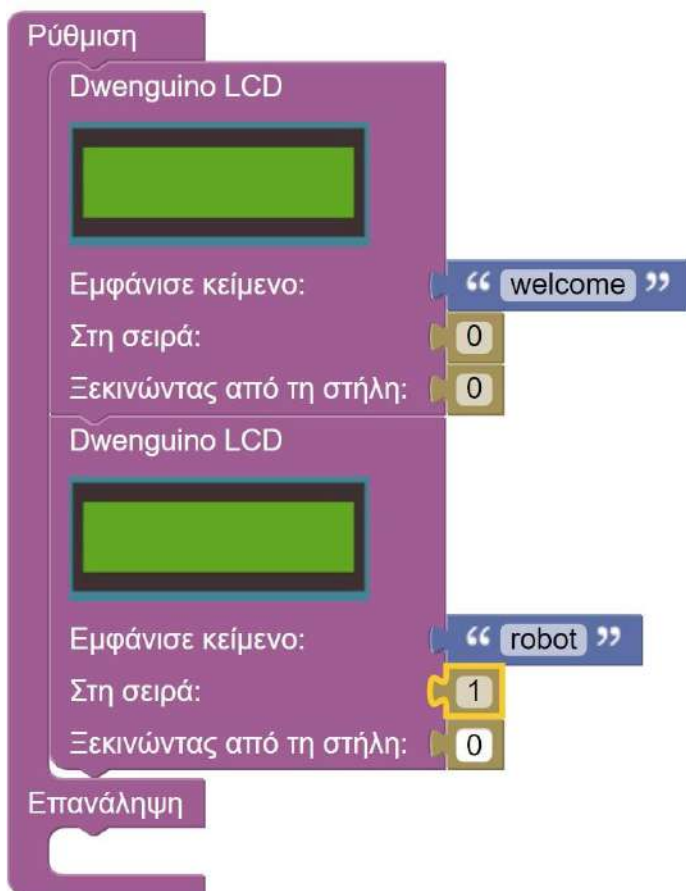
Also test these examples yourself in the simulator! Once you get the hang of how things work, you can get started yourself.

Example LCD display

SOLUTION 2.

Show 'Welcome' and 'robot' on separate lines.

Solution:



The image shows a Scratch script for controlling two LCD screens. The script is organized into three sections: 'Ρύθμιση' (Setup), two 'Dwenguino LCD' blocks, and 'Επανάληψη' (Loop). The first 'Dwenguino LCD' block is configured to display the text 'welcome' on the first line (row 0) starting from the first character (column 0). The second 'Dwenguino LCD' block is configured to display the text 'robot' on the second line (row 1) starting from the first character (column 0). The 'Επανάληψη' section is currently empty.

To split the text into 2 rows, you need a second 'lcd screen' block. If you change the 0 to a 1 at 'on row:', the text will appear on the second line.

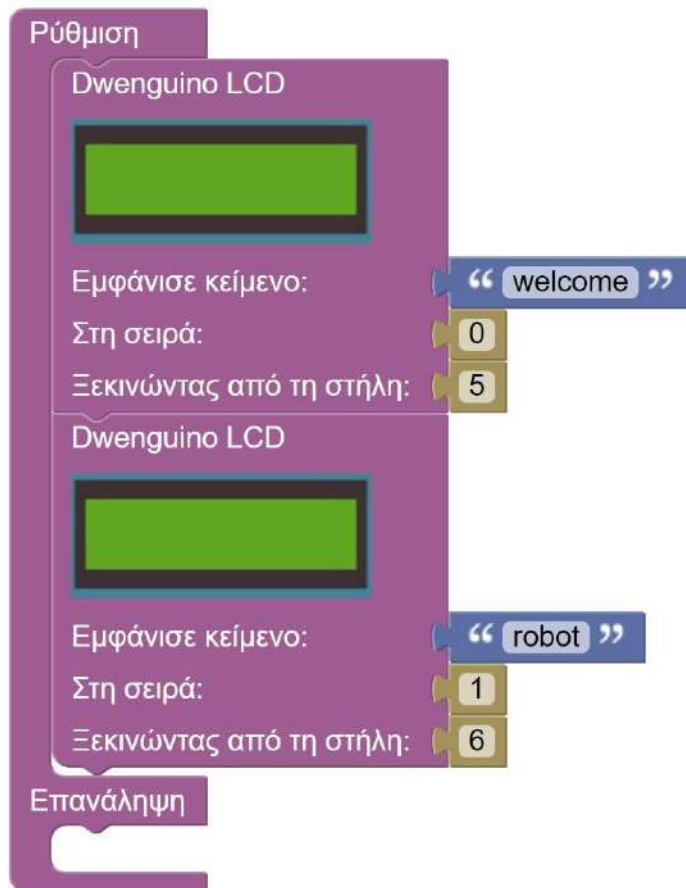
Test these examples in the simulator! Once you get the hang of how things work, you can try to display other things.

Example LCD display

SOLUTION 3

Now place the text in the center of the LCD screen.

Solution:



If you change the 0 to a 5 under "on column," the text moves 5 places to the right.

Wait

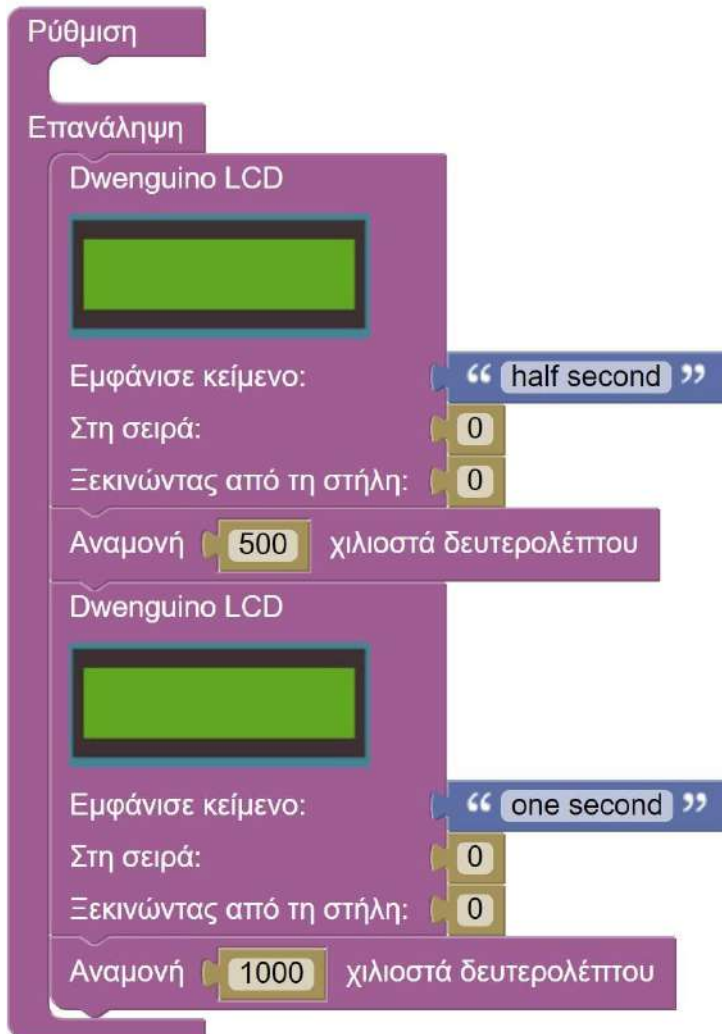
The 'wait' block is an instruction that lets the computer know how long something needs to run.



The time is expressed in milliseconds. One millisecond is one thousandth of a second. In one second, therefore, a thousand milliseconds can be used.

Example

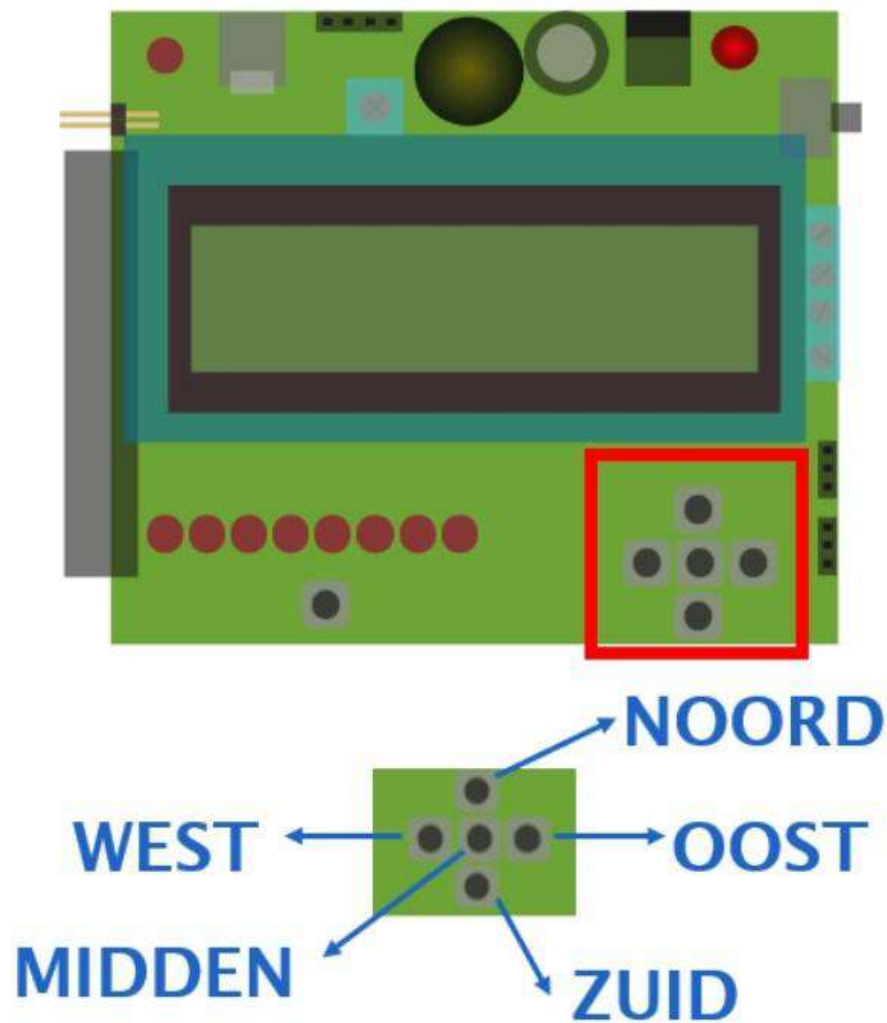
Try the example below so you can see how it works!



Butons

On the dwenguino you will find five push butons. The outer butons are named NORTH, SOUTH, EAST, WEST, just like in geography. The middle buton is called MIDDLE.

You can click on a buton with your mouse. When you click on it, the buton is pressed. When you release the buton, it is no longer pressed.



Obviously, you want the drawing robot to start drawing only when you want it to. Therefore, here you are going to enter an additional start condition as in the example below:

```

    Ρύθμιση
    Dwenguino LCD
    [LCD Icon]
    Εμφάνισε κείμενο: " I am a "
    Στη σειρά: 0
    Ξεκινώντας από τη στήλη: 0
    περίμενε μέχρι να πατηθεί το κουμπί Δύση
    Dwenguino LCD
    [LCD Icon]
    Εμφάνισε κείμενο: " drawing robot "
    Στη σειρά: 0
    Ξεκινώντας από τη στήλη: 0
    Επανάληψη
  
```

Example buttons

SOLUTION 1.

The program starts up. When the NORTH button is pressed, the first LED (LED 0) lights up.

Solution:

```

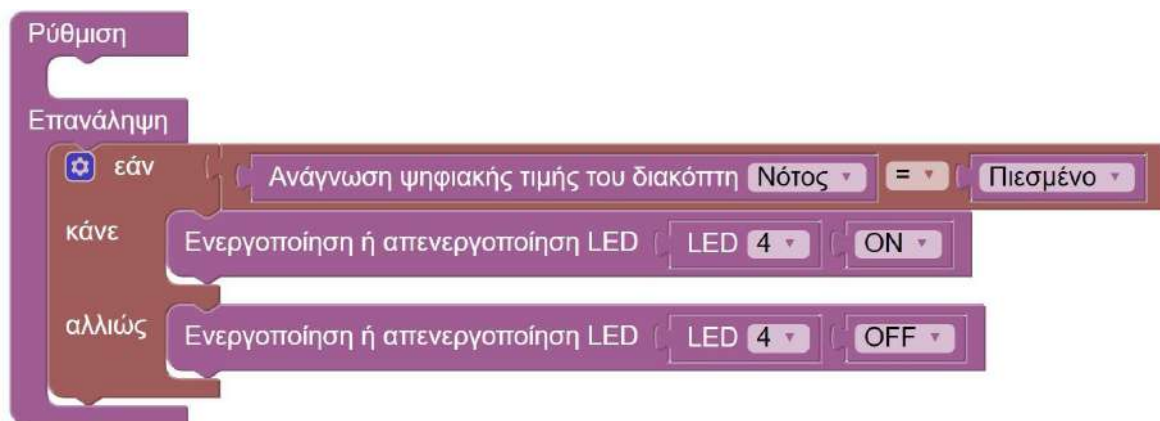
    Ρύθμιση
    περίμενε μέχρι να πατηθεί το κουμπί Βορράς
    Επανάληψη
    Ενεργοποίηση ή απενεργοποίηση LED LED 0 ON
  
```


Example buttons

SOLUTION 2

When the SOUTH button is pressed, the fifth LED (LED 4) turns on. When the button is released, the LED should switch off.

Solution:



Controlling the drawing robot

The WeGoSTEM's drawing robot has two motors and two arms. Each arm is connected to one of the motors. When these motors turn, the drawing robot's arms move. Each motor can turn right and left and can be set to different speeds.

Controlling a motor is done with a "motor" block:



With this block you control the speed and the rotational sense of the motor.



The lower red dots show where the motors are. The motors each have a number: the left motor is motor number 1 and the right motor is motor number 2. This corresponds to the number you see next to the text "number" on the 'motor' block.

The 'motor' block allows you to set the speed of that motor in addition to the number of the motor. That speed is set by a number between -255 and 255. With a positive value the motor rotates in one direction, with a negative value in the other direction.

Programming the drawing Robot

If you haven't started yet, it is finally time to experiment with your drawing robot yourself! Create the basic program and feel free to test out the things below!

Keep an eye on the time, so that you still have enough time to make a nice drawing after testing!

Task 1: Motors

Build the following program and see what it does.

Ρύθμιση

Dwenguino LCD



Εμφάνισε κείμενο: “ Let's draw ”

Στη σειρά: 0

Ξεκινώντας από τη στήλη: 0

περίμενε μέχρι να πατηθεί το κουμπί Βορράς ▾

Επανάληψη

Κινητήρας DC



κανάλι 1

ταχύτητα 150

Κινητήρας DC



κανάλι 2

ταχύτητα -255

Task 2: Testing

Let the drawing robot draw faster.

Have the drawing robot draw slower.

Run the motors of the drawing robot to the other side.

Task 3: Extension

Run one of the motors to the right and the other to the left.

Run one of the motors and not the other.


Experiment with the speeds of the motors.

After this exercise, you will be able to control the drawing robot. You now understand that the drawing obtained depends on the rotation sense and the speed of the motors.

If anything is still unclear, [this video summarizes everything again](#).

Packing up

Rest assured, you don't have to start packing up immediately now that you've finished the exercises! Just keep trying different programs until your teacher lets you know it's time to finish.

First reset the dwenguino by clicking the cross  in dwenguinoBlockly. Here you follow the same process as when you upload a program. Upon a successful reset, 'dwenguino ;)' will appear on the LCD screen.

Next, take everything apart and tidily put everything back in the box. This way, the next class can have a smooth start.