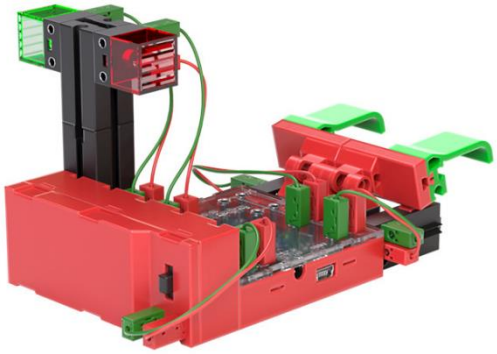
Smart Robots Pro

Getting Started



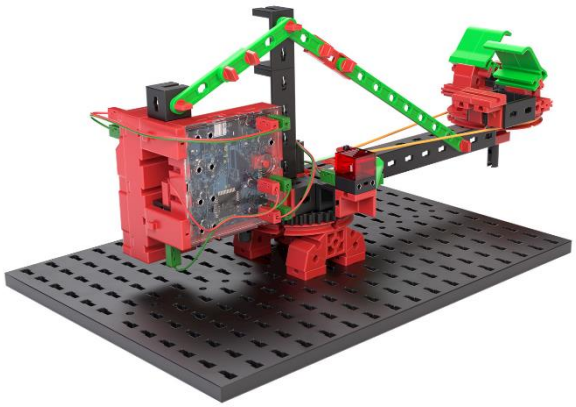
For your first steps in the world of Smart Robots Pro, we recommend that you watch the introductory video. Among other things, you will learn how to connect to the controller, how to perform an interface test and how to configure the controller.

Then you can start programming.

It is worth building and programming the models in the order shown here. You will gradually get to know new programming blocks that you will need again in later tasks.

Have fun!

Carousel



Svenja was at a funfair last week and was able to test many different fairground machines. Her favorite was the carousel. She couldn't get enough of the rides. So Svenja decides to build her own carousel. Can you help her assemble and program it?

- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth interface.

Task 1:

Once the controller is connected, you can carry out the first test drive using the interface test.

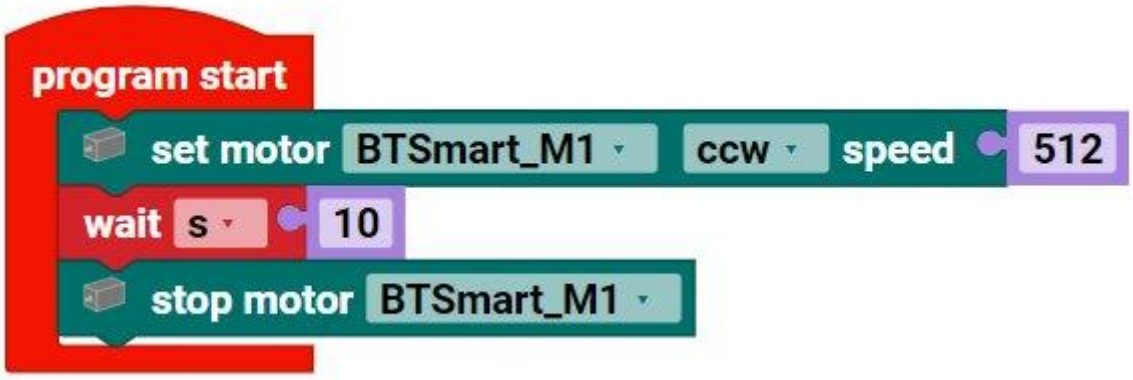
To do this, simply move the controller of motor 1 to the right and watch the carousel turn. You can also make the carousel turn in the other direction by changing the direction of rotation to the left.

Tip: If you're not sure where to find the interface test and how it works, it's best to watch the introductory video again.

The carousel has built-in lights, which you can switch on and off in the interface test via the controller of motor 2. What happens if you change the "direction" of the second output (M2)?

Task 2:

Svenja has already found a template for a carousel program on the Internet. However, she is not quite sure what the program does. Can you see what the illustrated sequence does in your carousel?



Drag the corresponding blocks into the programming field to create this program. Then test it.
To do this, the controller configuration must be carried out in Robo Pro Coding. If you do not know where to find this, it is best to watch the introductory video again.

Solution: The motor is started, runs for 10 seconds and is then stopped.

Task 3:

Svenja is afraid that the passengers could hit their heads due to the fast start. So she has an idea: the carousel should only travel at half speed (256) for the first 3 seconds and then accelerate to the maximum speed (512). Can you help her write such a program?

Task 4:

To make it easier to recognize whether the carousel is in the slow or fast stage, the green LED should light up at the slow speed - and the red LED at the fast speed. Can you add this function to your existing program?

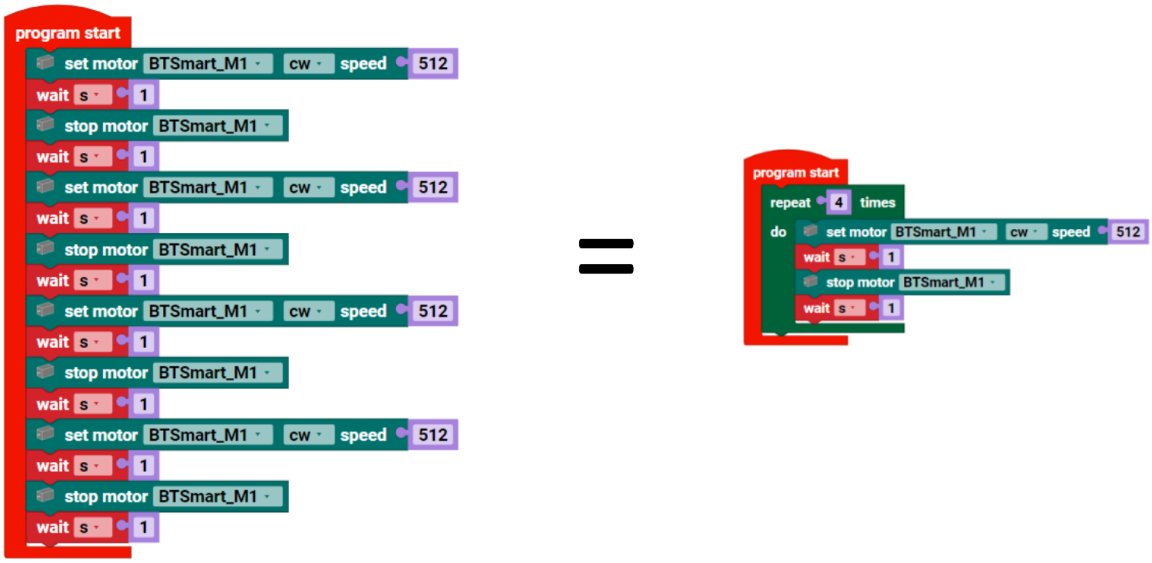
Tip: LEDs only work in one "direction". This means that the LED lights up when the current flows from + to -. However, it does not light up when the current flows from - to +. We can make use of this property so that we can control two lamps with one output. To do this, we change the LED to a motor in the controller configuration, as this allows us to change the direction of rotation, i.e. the direction

in which the current flows. If the motor input is set to counterclockwise rotation, the green LED lights up and if it is set to clockwise rotation, the red LED lights up. First try out this function in the interface test. You can then incorporate it into your program.

Task 5 (advanced):

Before the carousel starts moving, the red LED should flash three times. After the fast phase, the speed is halved again for three seconds. The direction of rotation of the carousel then changes, also with a 3-second slow phase and a 7-second fast phase. The ride ends with another 3-second slow phase.
The red LED lights up at slow speed and the green LED lights up at fast speed.

Tip: When programming, contiguous commands are often required several times in succession. To avoid having to program an endless number of blocks, you can also use so-called loops. Here you can see two programs that have the same effect, namely that output M1 is switched on and off (or flashes) 4 times in succession.



You can find sample solutions for the model in Robo Pro Coding.

Useless Machine

Nik's little brother Ole sometimes grabs Nik's sweets. So he decides to build himself a little box that only opens very briefly. As Ole is not yet so quick, Nik can take the sweets from the box, but Ole cannot.
It's also fun to use and watch the box.
Will you help Nik to build and program such a box?

- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth or USB interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

Task 1:

You can use the interface test to move all motors to the starting position. Check whether the buttons are open or closed.
Try using the interface test to open the snout of the box, extend and retract the operating lever and then close the snout again.

Tip: It may be that the operating lever does not immediately switch off the switch with the power of the motor. In this case, simply operate the switch with the black swivel joint 10 times by hand all the way to the front and back and check whether your battery is still supplying sufficient voltage.
Avoid running the motors on block for long periods of time, i.e. trying to turn but not being able to do so because they stop.

Task 2:

Now create a program for the Useless Box:
When button I4 is pressed, the snout of the box should open and stop after 0.6 seconds. The operating lever then extends and opens button I4 again. The operating lever then moves back to the starting position so that button I2 is closed. The snout then closes again.
You can use the wait until block for this program. This stops a program until the attached condition is fulfilled and only then continues.

You can find sample solutions for the model in Robo Pro Coding.

Dance robot

Clara loves carnival and parades of all kinds. However, her hands always get so heavy from all the waving. So that she can still greet the many beautiful floats in the town parade, she simply builds a robot that waves for and with her.

- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth or USB interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

Task 1:

Use the interface test to change the position of the arms so that the left arm is pointing vertically downwards.
Program the robot so that it waves continuously by moving the arms in a loop for 2 seconds at a speed of 100, then leaving the arms stationary for one second and moving the arms in the other direction again for two seconds. After the two seconds, the arms remain stationary for another second.
Make sure that you move the arms back to the starting position with the interface test before each program start and that the motors do not block.

You can vary the speed as you wish - then also adjust the duration.

Task 2:

To draw even more attention to her dancing robots, Clara is thinking about making the robot nod after every second wave by having the cam disk rotate counterclockwise at a speed of 350 for 2 seconds after every second wave. Can you help her write a program to do this?
Make sure that you move the arms back to the starting position with the interface test before each program start and that the motors do not jam.

You can find sample solutions for the model in Robo Pro Coding.

Drivebot

- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

Task 1:

Program the robot so that it moves straight ahead after the program starts. If the robot encounters an obstacle, i.e. if one of the two openers is activated, the robot moves backwards for one second and then turns by moving one wheel forwards and the other backwards for one second. The speed of the edges when turning is 200 for two seconds, after which the robot continues straight ahead again.

Push-buttons can be used as normally closed or normally open contacts. The state of a normally open contact is 0 when unactivated and 1 when activated. The opposite is true for normally closed contacts: unactivated is 1 and activated is 0.

Tip: Make sure that the joints of the fire buttons are pushed all the way up so that the buttons are actuated reliably.

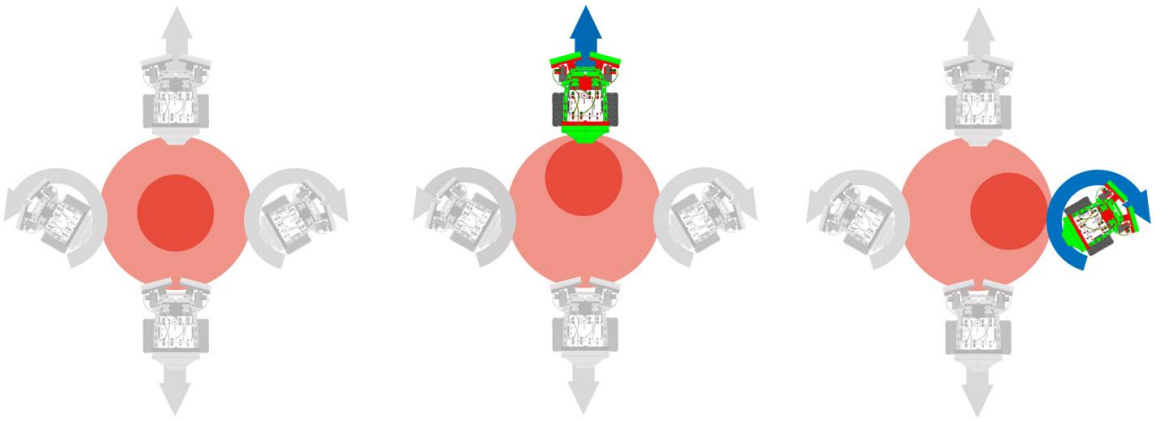
Task 2:

Complete the program so that the robot corrects its path to the left if it encounters an obstacle on the right and vice versa, corrects its path to the right if it encounters an obstacle on the left.

You can find sample solutions for the model in Robo Pro Coding.

Joystick - Remote control

Now we are going to write a small program with which we can easily control our driving robots remotely. The program contains a digital joystick. This joystick can be dragged back and forth on your touch display or on the computer. Our driving robot should then drive in the direction in which we drag the joystick. For example, it will drive straight when we pull the joystick upwards and it will turn when we pull it to one side.



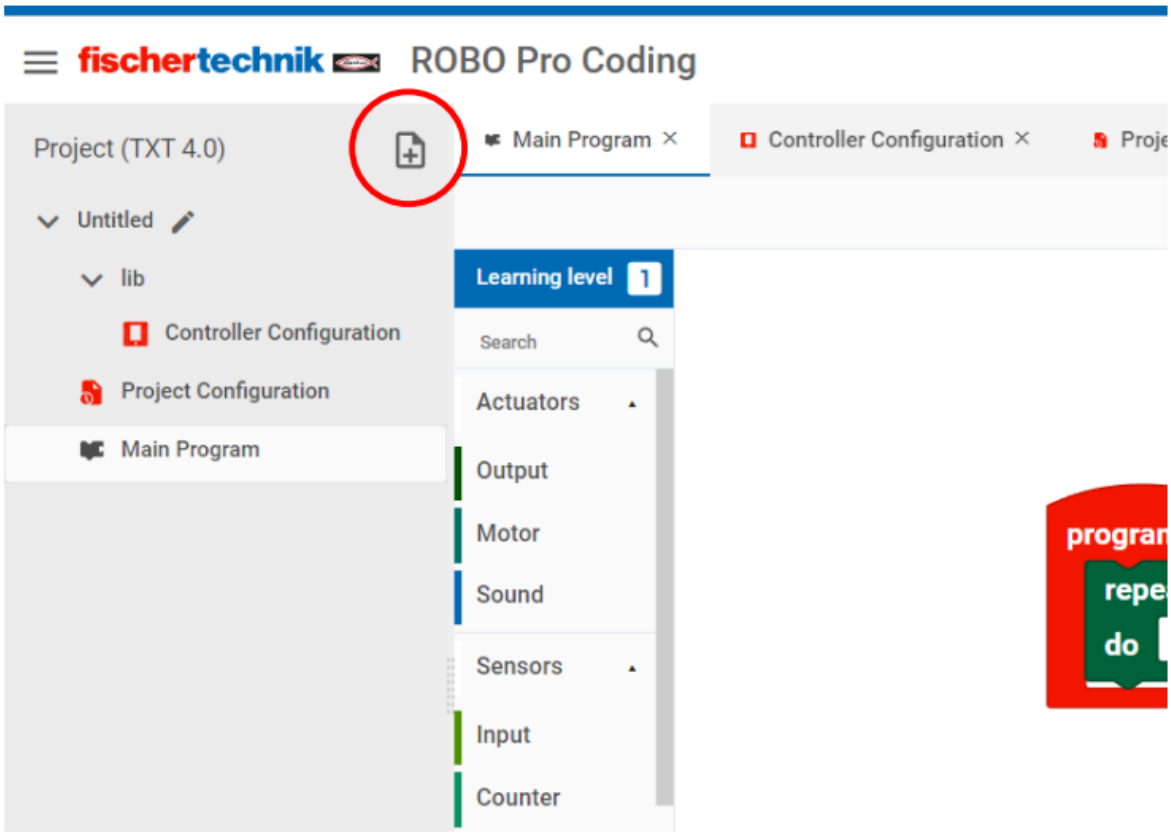
You can use the programmed joystick to control the Drivebot or the Teach-In robot remotely.

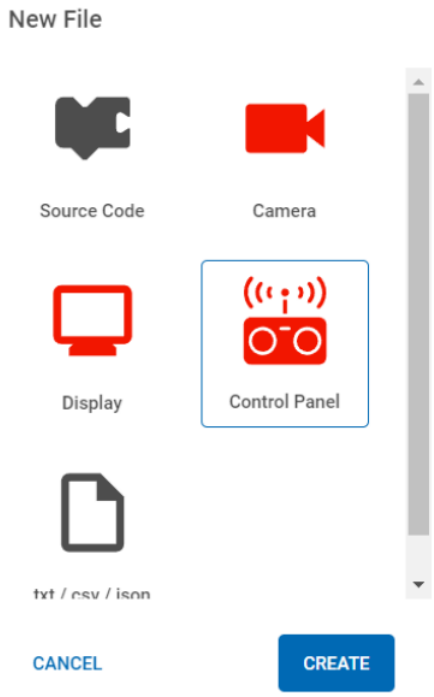
If you want, you can also test what we are going to program first. Simply open the sample program "BTSmart_RC_Joystick_5" and test the program.

In the following paragraph you will get to know some new functions of Robo Pro Coding, which we also need for other robots.

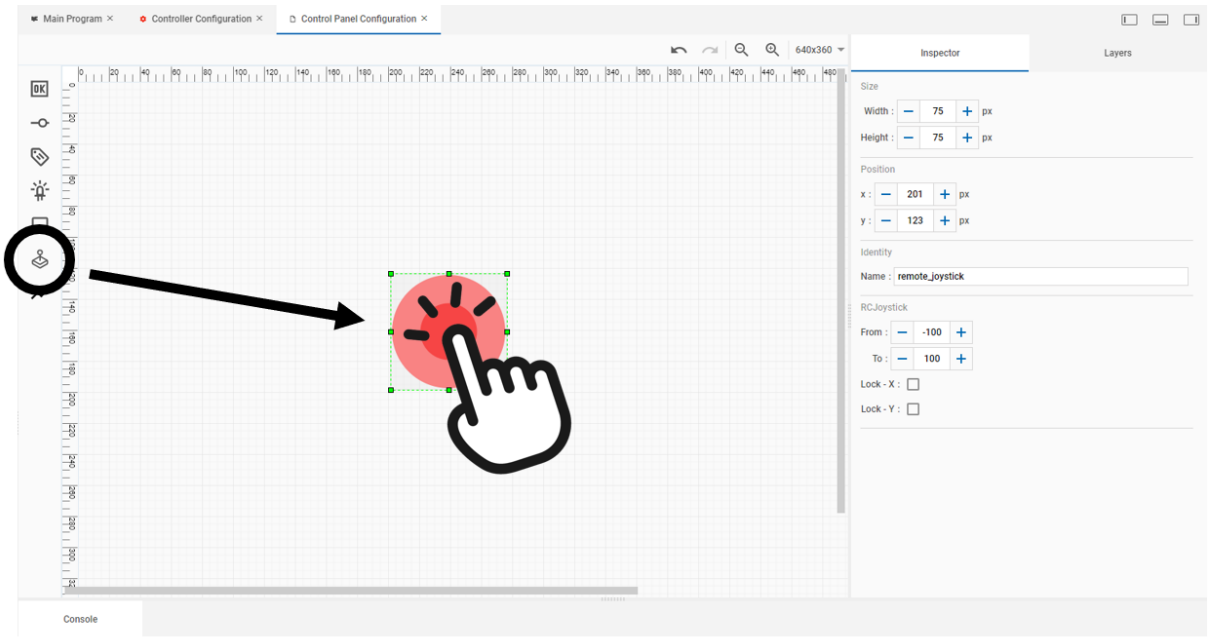
- Build any driving robot model
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

Before we start writing the program, we need an additional button for the joystick. To do this, click on **New file** and then on **Control panel.** This is added by clicking on **Create.** You can see these steps in the following pictures:





A new empty interface will now appear. We will see this interface on our computer or tablet when we start the program. First, we drag the so-called RC joystick onto this interface:



You can use the green dots on the corners to make the joystick larger or smaller. We have now created a joystick.

Test it out by connecting your controller and starting the program in Robo Pro Coding.

As you can see, when you start the program, the previously drawn window opens and you can drag the inner part of the joystick back and forth.

Using the joystick, we can now write a program that makes the driving robot move in the corresponding joystick direction. To do this, we switch back to the programming environment.

Variables

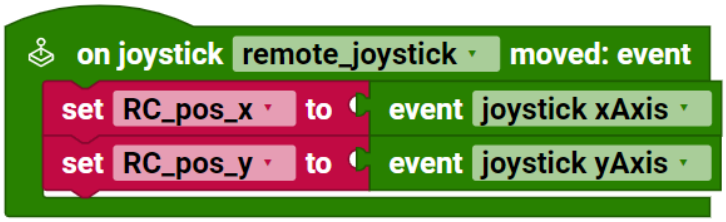
Variables are memory blocks. We can store texts or numbers in these blocks and use them later in our program. We now use variables to store the position of our joystick.

To do this, we first switch to learning level 2 and create two new variables under Variables:

We name these variables "RC_pos_X" and "RC_pos_Y". We can now access these memories at any time under the Variables tab.

There is a joystick event block under the remote control field, which we can use to always write the current values of the joystick position to our variables.

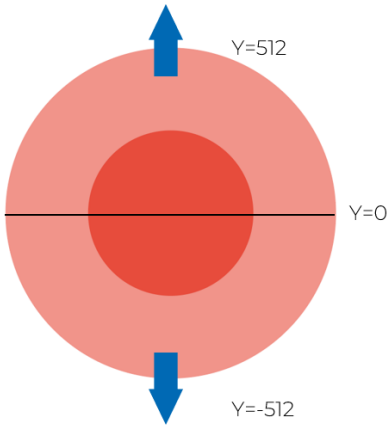
Use these blocks to create a program that saves the current position in the respective variable each time the joystick position is changed:



The Y values of the joystick change when it is pulled up or down. The X values change when it is pulled to the left or right.
In its initial position, it has the values X=0 and Y=0.
The further the joystick is pulled upwards, the higher the Y values become.
The further you pull the joystick down, the smaller the Y values become (they become negative values).

Task 1:

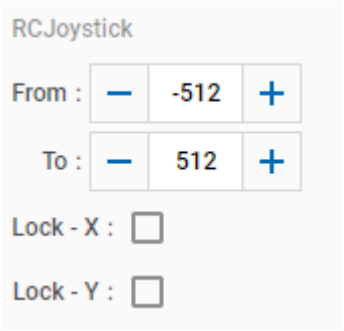
Write a program that sets both motors to full speed if the variable RC_pos_Y is greater than 0. If the variable is less than 0, your driving robot should drive backwards with both motors turning in the other direction. To test whether the variable is greater or less than 0, you can drag a comparison block from the Logic tab and a number from the Math tab.



If you are not sure how to do this, you can take a look at the example programs.

Task 2:

In the next step, we also control the speed of the driving robot using the joystick. To do this, we switch back to the control panel configuration and select the joystick we have drawn. In the right-hand window, we now find setting options for the joystick. To make it easier to control the speed, we extend the range from -512 to 512.



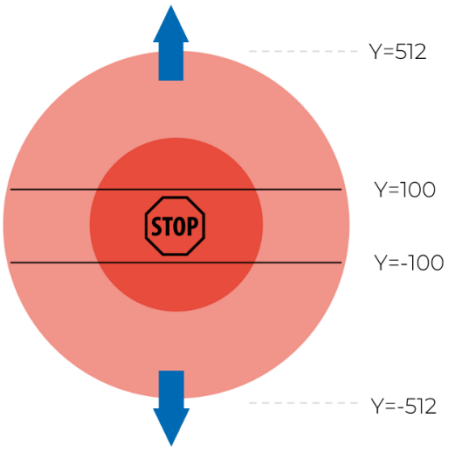
Now write a program in which the driving robot always drives as fast as the value of the variable RC_pos_Y is large.

Tip: Up to now, we have always defined the speed of the motor blocks with fixed values. However, as we have stored numerical values in our variables, we can also append the variable to the block. To do this, simply drag the variable block over the number. This is automatically replaced.

Tip: When using variables, the motor can also accept negative numerical values. This means that if a value is less than 0, e.g. -256, it automatically moves backwards and you do not have to change the direction of rotation of the motor. You can therefore theoretically write this program without an IF block.

Task 3:

As the voltage is not sufficient to move the driving robot at very low values, we introduce additional zones. Namely, the driving robot should only start moving when the value of the Y variable is greater or less than 100 / -100



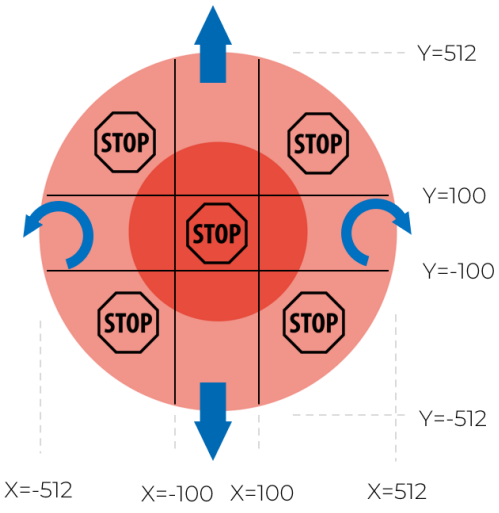
To do this, we need another query that we are getting to know: the OR link:

With the If blocks, we have only compared one condition so far. With the OR blocks, we can also query several conditions. With the OR block, one of several queries must be correct for the instruction to be executed. With the AND block, all queries must match for the instruction to be executed.

Extend your program from task 2 so that the driving robot only starts moving when the variable RC_pos_Y is greater than 100 OR less than -100.

Task 4 (advanced):

Now we introduce additional zones so that we can also rotate the driving robot. To do this, we now also use AND links.



Extend your program so that the driving robot moves at the speeds of the variable RC_pos_Y if the following conditions are met:

RC_pos_X is less than 100 AND greater than -100.
At the same time (AND), RC_pos_Y must be greater than 100 OR less than -100.

➔ These conditions should control forward and reverse travel.

If RC_pos_Y is less than 100 AND greater than -100
AND at the same time RC_pos_X is greater than 100 OR less than -100,
the motors should rotate at the speed RC_pos_X. Reverse the direction of rotation of a motor.

➔ These conditions are intended to control clockwise and counterclockwise rotation.

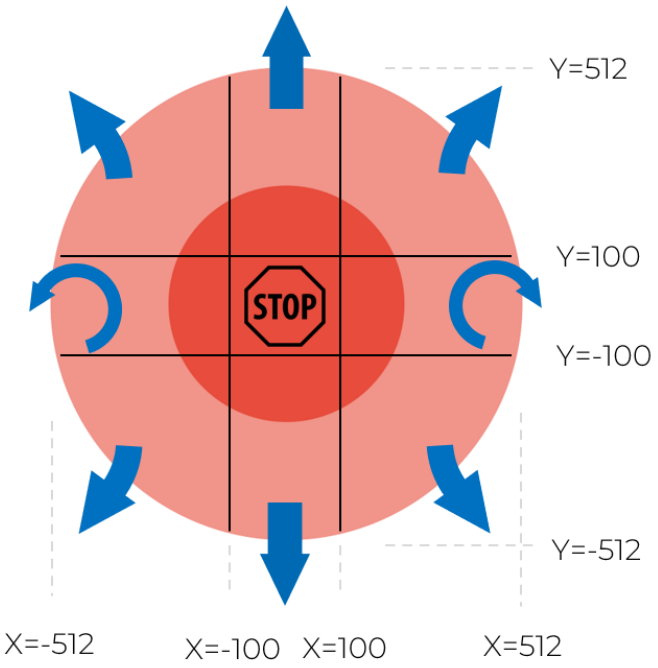
Task 5: (expert task):

If you pull the joystick at an angle, the robot will stop.

Now write a program so that the robot makes a large turn when the joystick is in one of the four remaining fields. To implement this, the motors should move in the same direction, but one motor should move more slowly than the other.

It can be helpful to create a few more variables. However, this is not absolutely necessary. Try to write your own individual control program.

The following diagram can help you to determine the remaining zones:



Let's move on!

Congratulations! You are now an experienced programmer. Don't worry if you have skipped the advanced or expert tasks. The first remote control tasks have still taught you many new programming blocks that we can use to create further programs.

If you need a break before the next programming session, use the teach-in robot, for which we simply use a ready-made program:

Teach-in robot

Teach-in robots can be taught what to do directly on the robot using inputs. They are complicated to program, so you can directly open the example program in Robo Pro Coding.

- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

Task 1:

Build an obstacle course. Then try to master it with the teach-in commands. The button input works more reliably if you press it for several seconds per command.

Button left: Left rotation (90°)
Button right: clockwise rotation (90°)
Both buttons simultaneously: straight ahead

By default, the robot moves after 5 commands. However, you can also change the number by changing the value of the variable list_move < 5. You can also see what happens if you change the speeds.

Alarm system

More than 65,000 homes were broken into in Germany in 2022. That equates to 178 break-ins per day. It's good if you know how to help yourself and protect yourself with technology. To find out how an alarm system works, we build one ourselves.

- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth or USB interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

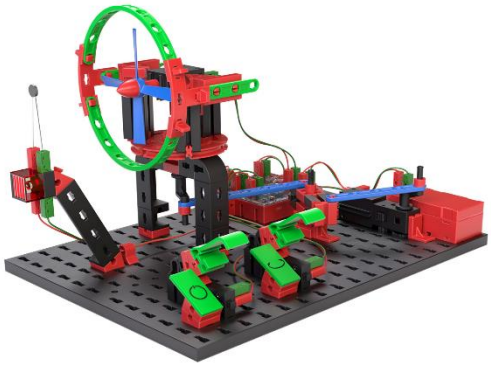
Task 1:

Program the alarm system as follows:

- If the main switch is not activated, nothing should happen: So you and your family can open and close the door as normal.
- If the main switch is actuated and the door is closed so that the reed contact has status 1, the green LED lights up.
- If the main switch is activated but the door is open, the red LED flashes and the rattler makes a noise for one second every two seconds.
- If the door is then closed again, the alarm should still be triggered.
- The alarm only stops when the main switch is switched off again.

You can find sample solutions for the model in Robo Pro Coding.

Fan



Fans are very simple but useful machines. A simple fan can be switched on and off and can also be made to move back and forth. We now want to build an intelligent fan that starts these two functions automatically.

- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth or USB interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

Task 1:

Program the fan so that it can be rotated and moved back and forth using the two switches. The fan should only move back and forth when it is rotating.

Task 2:

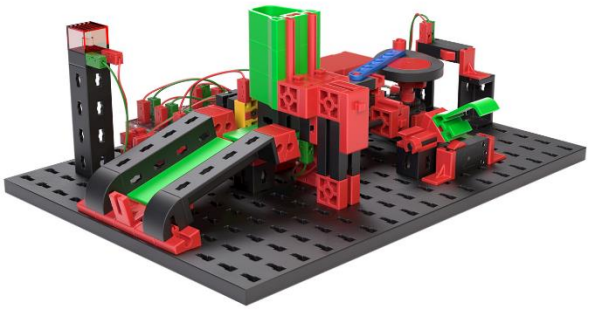
Make the fan control intelligent by starting the fan automatically as soon as the NTC resistance value is less than 1300. If the value is exceeded, the fan can still be controlled via the switches.

To do this, we let the controller calculate. To do this, you need to switch to learning level 2. Here we compare in a logic block whether the sensor value is smaller (<) than a specified value.

Tip: You can adjust the values if necessary. Check the interface test to see what value the NTC resistor outputs in the "cold state". You can simulate the heat by touching the NTC resistor with your fingers.

You can find sample solutions for the model in Robo Pro Coding.

Component dispenser



- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth or USB interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

Task 1:

Program the component dispenser so that it dispenses a component each time you press the button (I1). If no more components are detected in the magazine, the signal LED goes on and the dispenser no longer responds to the button press.

Task 2:

Turn the component dispenser into a component counter. After the trigger is pressed, the counter ejects component after component from the magazine until it is empty. The components should be counted in parallel and the result output as text.

You can use variables for this. You can create a variable with the name "count" and set it to 0 when the program starts. With each block that is ejected, you can add +1 to the variable.

You can find sample solutions for the model in Robo Pro Coding.

Painting robot

- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth or USB interface.
- Use the interface test to check whether all electronic components are connected correctly.

Task 1:

Grab a sheet of paper and stick it to the table using two adhesive strips. Take a pen that fits into the pen holder of the painting robot and attach it to the holder. Then align the painting robot with the paper.

In the interface test, you can now create countless patterns by changing the motor speeds and directions of rotation. If you particularly like a pattern, you can also create a program with the selected parameters so that you can draw it again and again. Let the painting robot draw for a while until you can recognize a pattern.

Always use a paper when using the painting robot and make sure that the paper does not let any paint through.

Task 2:

Another influence for the creation of the patterns are the axes of rotation. Try what happens when you change motor positions.

Ball game

- Build the model according to the online building instructions.
- **Rebuild the model with the additional optimization steps.**
  [You can find these online]

Ball game

To the building instructions

Optimization model

- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth or USB interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

The battery used should have a voltage > 8.5V so that the throwing mechanism functions reliably.

A ball is thrown by the motor tensioning and releasing the strut during a rotation. Another turn reloads a ball from the magazine

If button I2 is pressed, the motor M1 rotates until button I1 is closed.

Task 1:

Create a program that triggers a throw each time a button is pressed. Make sure that the direction of rotation is correct for the throw. Abort the program if the direction of rotation is not correct and the motor stalls.

In the interface test, you can adjust the inclination of the ejector by controlling the second motor.

Task 2:

To make hitting the target a little harder, you can also change the inclination continuously. To do this, simply add to your program that motor 2 runs continuously at a low speed.

You can find sample solutions for the model in Robo Pro Coding.

Swing boat

- Build the model using the assembly instructions.
- Connect the cables according to the wiring diagram.
- Start the ROBO Pro Coding software.
- Connect the BT-Smart Controller to the computer or another mobile device via the Bluetooth or USB interface.
- Use the interface test to check whether all electronic components are connected correctly.
- Then carry out the controller configuration.

When building the model, make sure that the ship just rests on the tire. Then swing the swing by hand and use the interface test to ensure that the reed contact triggers when the magnet on the ship's swing passes the reed contact.

Task 1:

Create a program that accelerates the boat swing by rotating the tire. As soon as the reed contact is closed, the tire should rotate in the other direction for approx. 0.2 seconds. The direction of rotation then changes again until the reed contact is closed again.
The program should only start when the main switch (I1) on the model is closed. As soon as the main switch is opened again, the swinging ship should swing out again, actuate the reed contact once more, swing back in the other direction and then stop so that the ship is upright against the tire.

Tip: To ensure that the tire accelerates the boat swing reliably, you can push it to the left or right with the motor. When swung out, the tire should always touch the swing boat.

Task 2:

Create a variable with the name "rounds" and save the value 4 in it. Before the ride starts, the LEDs should flash alternately, namely as often as the value rounds is large. Afterwards, the boat swing also swings as often as the value round is large.

Are you able to assign a random value between 1 and 6 to the variable?

You can find sample solutions for the model in Robo Pro Coding.