WORLD ROBOT OLYMPIAD ™

WRO®

SWITZERLAND

ADVANCED ROBOTICS

VARIABLES AND MyBLOCKS

© 2023Verein World Robot Olympiad Schweiz
Offizielle Ausrichterin der World Robot Olympiad in der Schweiz

Version 1.1_EN
01.03.2023 / vh

INHALTSVERZEICHNIS

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

1 INTRODUCTION: LOOPS

There is one golden rule for programmers:

> ***"Do not write anything twice when you can write it just once."***

Let us look at a very simple example:

You want your robot to drive a square figure. Without our golden rule, our code will look like this:

Now, if you realize in your tests that the block "**Move right: 100 – 135 degree**s" does not result in a right angle but is either a little too much or not quite enough[1], then you need to fix it in four places. This way, it's very easy for mistakes to slip in which will later be hard to find and fix.

[1] Which value you need depends on,
- The size (diameter) of your wheels
- The distance between your wheels
- How fast the robot is driving (due to inertia, when you drive fast you will need a slightly smaller number than when you drive slowly.).

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

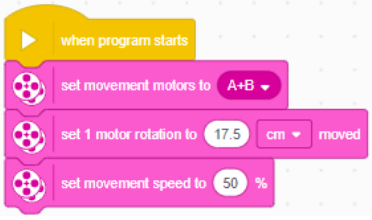
Therefore, it makes sense to analyze first which parts of the code occur more than ones, and test those first, without bothering with all the rest.

In our square shape, the part that occurs multiple times is this:

We can just plug these two lines directly under the header of the code[2] and test. If necessary, we can fix any errors. When we are satisfied with the result, we can repeat these two lines with the help of the **loop** block. As you see, our code is already much shorter than it was before:

Header with the definition of drive motors

[2] The header of the code is the part that stays the same in all driving programs:



If you use the standard wheels included in the SPIKE Prime box, have the drive motors assigned to ports A and B, and want to drive at 50% speed, you don't need these blocks. But it is best practice to include them anyway, as you might one day not be using these standard configurations.

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

2 VARIABLES

A **variable** is a value that is used in programming. A variable can be used multiple times. And a variable can change its value during the execution of the code.

This sounds complicated, but it isn't. Let us look at our square shape to understand better, what this means:

We want the light matrix on our robot to display which side of the square shape the robot is driving at any given moment. So, while it's driving along the first side, it must show a "1", along the second side a "2", and so on.

For displaying text or numbers on the light matrix, you need the "**write**" block.

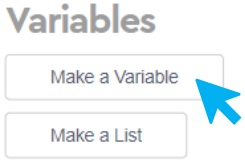
So, at the beginning of the first side, you could enter the "write" block like this:

But now our lovely loop would be useless because it repeats exactly what is inside. But we don't want to repeat the number "1" four times – we want the number to change! The rest of the code should stay the same.

This is where a **variable** is useful.

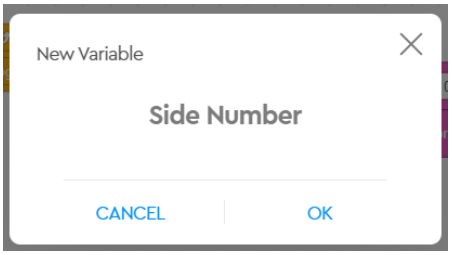
1. In the block list, go to the dark orange "**Variables**" section.

2. Click on "Make a Variable".

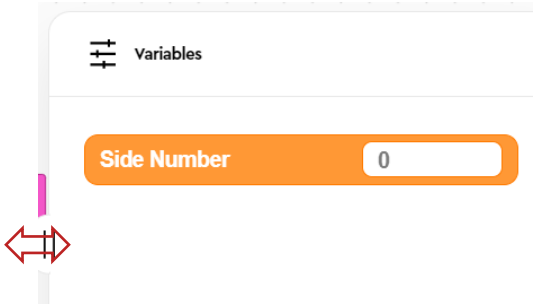


© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

3. Name your Variable "side number", then click OK.



4. On the right side of your screen you will now see a window in which all variables are listed that you have defined for this project. You can close this window so it is not in your way (especially if you have a small screen) by clicking on the "handle" at the left edge of the window as shown with the red double arrow. ⇔.



5. In this window you can watch how the value of the variable changes while the program is being run. Of course, your hub needs to be connected to your device for that to works. It's a useful tool to check if your program is working as desired.
6. In your block list you now see three new blocks that were not there before:

- **Side number**»: You can enter this block in any rounded input field. Here you see a few examples of what this might look like. Not all examples always make sense, but you see how this can work.

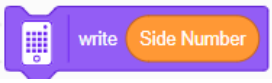
In every block where our "Side number" variable is entered in the input field, the current value of this variable will be used.

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

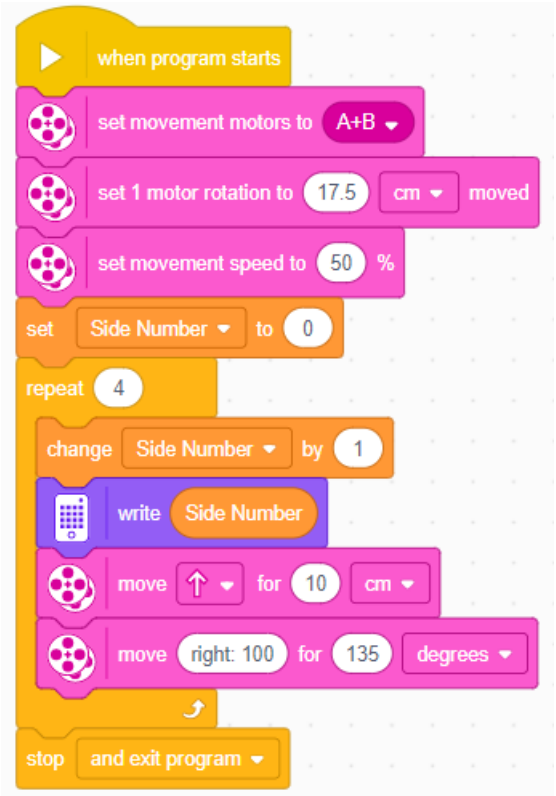
- "**Set side number to …**": With this block you give the variable a fixed value. Usually, it is the value the variable has at the beginning of the code. In many cases, zero will be the best value. This is also true for our square shape.
- "**Change Side Number by …**": We need this block for our side counter. It doesn't assign a new value, but it describes a rule what happens to the variable. "**Change Side Number by 1**" means: Add +1 to the value of the variable. So, if the variable has the value 0 at the beginning, this block adds 1, so the variable will then have the value 1.
  If you write a negative number into this input field, then that number is subtracted from the value of the variable. This is useful for a countdown.

7. After the header part of your code, place the block "**Set Side Number to 0**".
8. Now place the block «**Change Side number by 1**" just before your two blocks describing the side and the corner of your square shape.

9. After that, place a **write** block and put the variable "**Side number**" into the input field.



10. Now just place the whole thing inside the **loop**, and our code is finished!



© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

3 MyBLOCKS

Loops are wonderful if you want to repeat an action multiple times without anything else happening in between. But sometimes you need to use the same action more than once but not consecutively, as the robot will have to do other things in between.

Let's try this with our square shape from Chapter 2.

Our robot must now drive the same square as before, counting the sides on the light matrix, but after each turn, it has a different task:

- After the first turn, it must make three short, deep "beeps" (note number 60).
- After the second turn, if must display a happy face on the light matrix for one second.
- After the third turn, the main light must blink red three times.
- After the fourth turn, it must make three short and one long high beep (note number 72).

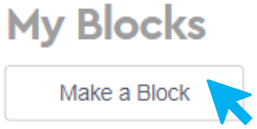
You will notice: These tasks are so different that we cannot put them inside a loop. Does that mean that we have to rewrite the code for the side and turn and side counter after each special activity? As we learned earlier, that is how mistakes happen! So we use a much cleverer way:

We can write the side and the turn and the side counter into a "**MyBlock**".

Du siehst: diese Aufgaben sind so verschieden, dass du sie nicht in eine Schleife packen kannst. Bedeutet das, dass wir den Code für die Seite und die Ecke wieder jedes Mal schreiben müssen, mit dem Risiko, dass sich Fehler einschleichen? Nein! Es gibt eine viel cleverere Lösung:

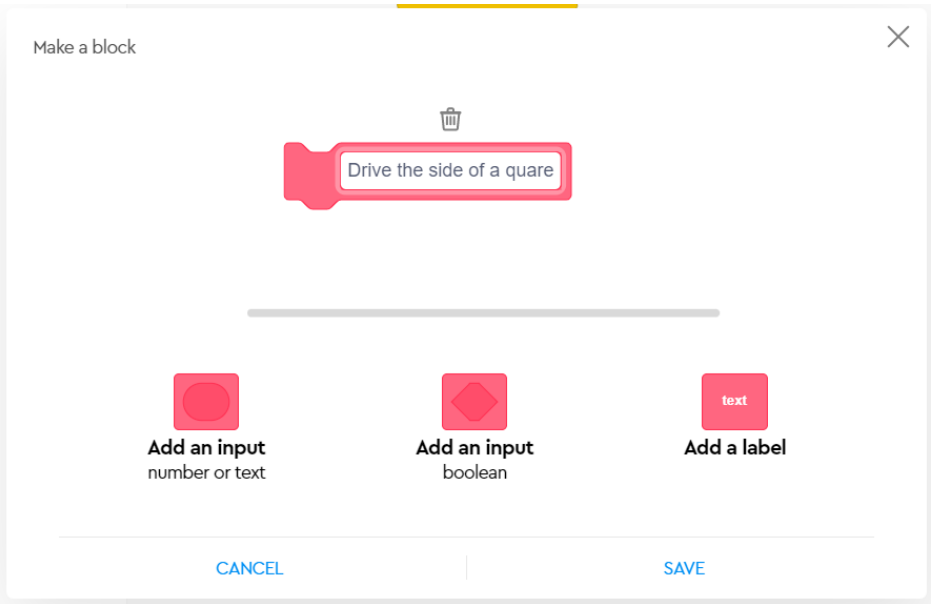
1. In the block list, go to the "**My Blocks**" section.

2. Click on "**Make a block**".



3. You will now see a window where you can give your new MyBlock a name. It is good practice to include a verb in this name. So, a good name here would be "Drive a side of the square". Write this and click OK.

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

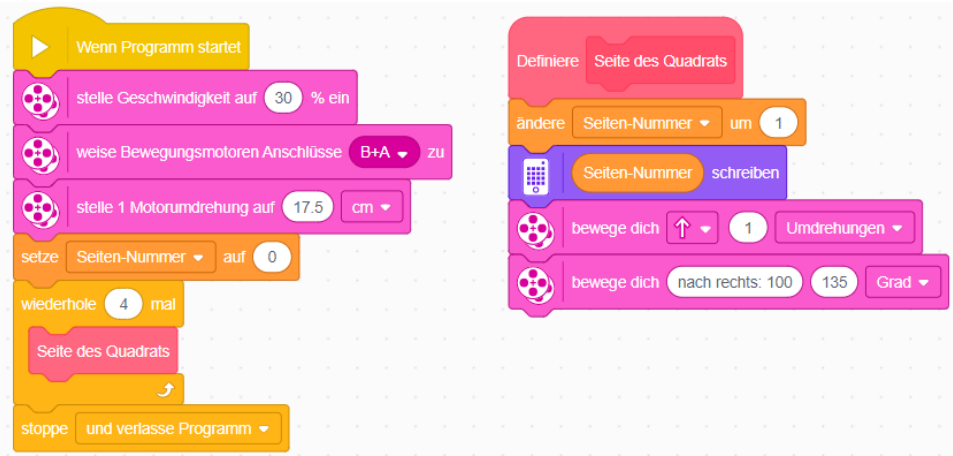


Below the name, you can add input fields for your MyBlock, where you can write values just like in other blocks. These are called parameter. For our example here we do not need parameters. At the end of the dossier, you will find a brief explanation.

4. On your screen you now see a new stack head with the name of your MyBlock:

And the MyBlock has also appeared in your Block list:

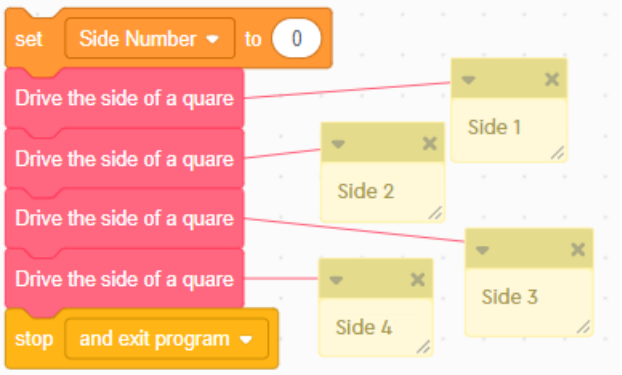
5. Now, drag all your blocks out of the loop and place them under the MyBlock head. For the time being, leave the empty look in your main stack so we can test that the MyBlock works as desired. Place the "Drive a side of the square" inside the loop. The code should now look like this:



© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

Now start your program. It should do the exact same thing as before when we didn't have a MyBlock.

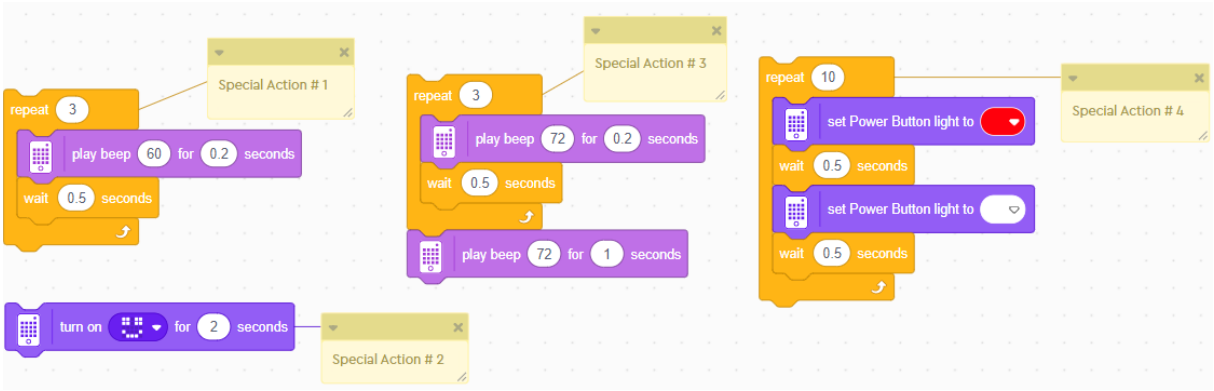
6. At the moment, the code looks much more complicated than before. Where is the use in moving the side counter and the driving into a MyBlock? Look again at the task set at the beginning of this chapter. After each turn, the robot must perform a different activity. So, the loop has become useless. Therefore, we delete the loop from our code.
7. Now we think about the logic of our program:
   We always need one side of the square with the side counter and a turn, then we need the special activity.
   We start by adding 4 "Drive a side of the square" MyBlocks under the block "Set Side number to 0". This gives us a scaffolding into which we can then build the rest.
   To make the structure easier to see we can add a comment[3] to each MyBlock:
8. Jetzt überlegen wir uns die erforderliche Logik unseres Programms:
   Wir brauchen immer die Seite des Würfels, dann die Spezial-Aktion.
   Am einfachsten fügen wir unter dem Block «Setze Seiten-Nummer auf 0» zunächst vier Mal unseren «Seite des Quadrats»-MyBlock ein. Und um es später ein bisschen übersichtlicher zu haben, können wir jeweils noch einen Kommentardranhängen:



9. Now write the code of one of the four special activities after each MyBlock. These code pieces look like this:

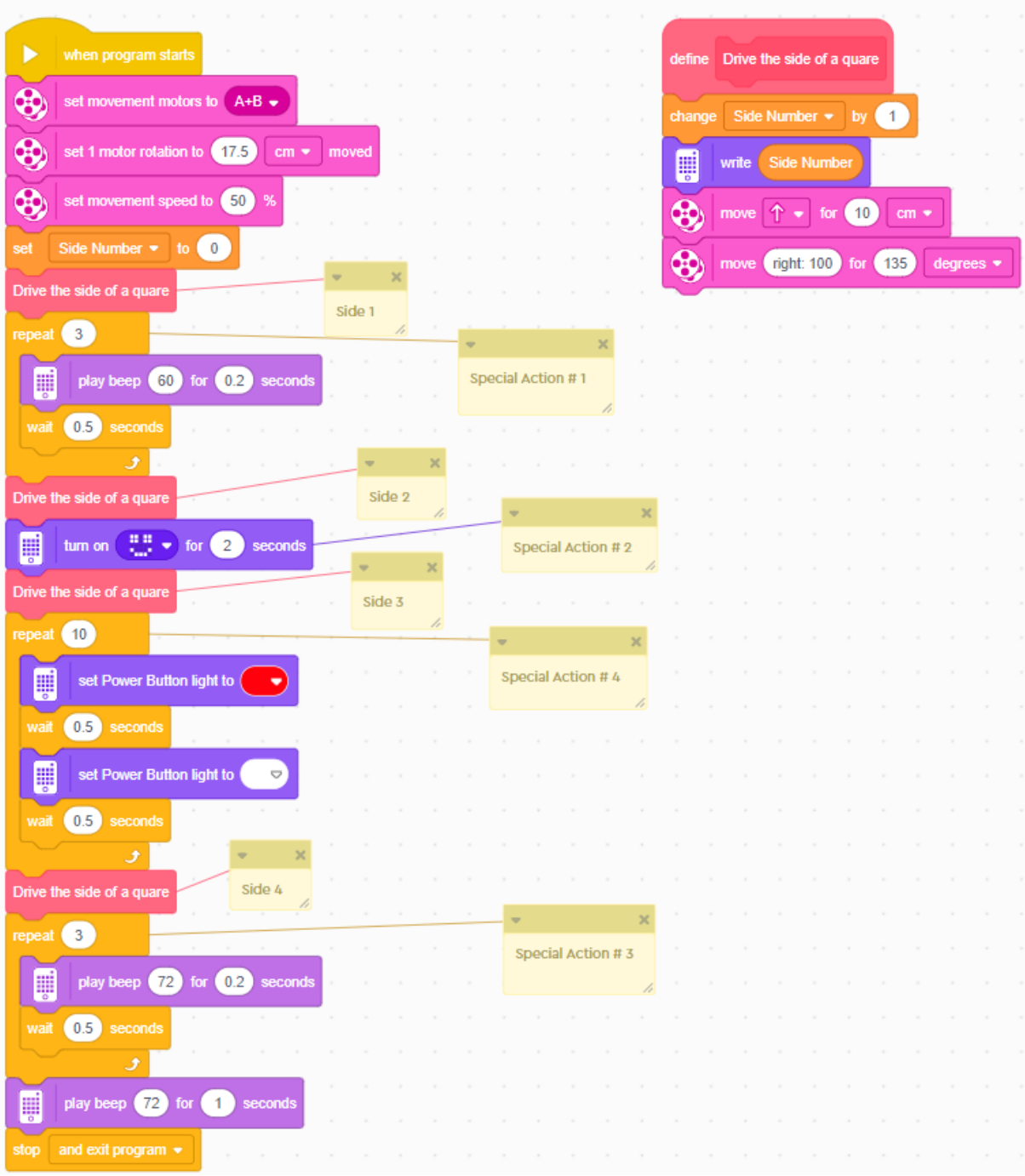
[3] It is best practice among programmers to comment code as much as possible. This helps you for clarity of the code so you will find your way around long and complicated code more easily. It also helps to remember what you were thinking while coding a specific section, or a partner to follow your thoughts. In the SPIKE App, right-click, or long-tap on the block, then select "Add comment" from the context menu. You delete a comment by clicking the X in the comment box.

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.



10. Done! Your program should now look like this:

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

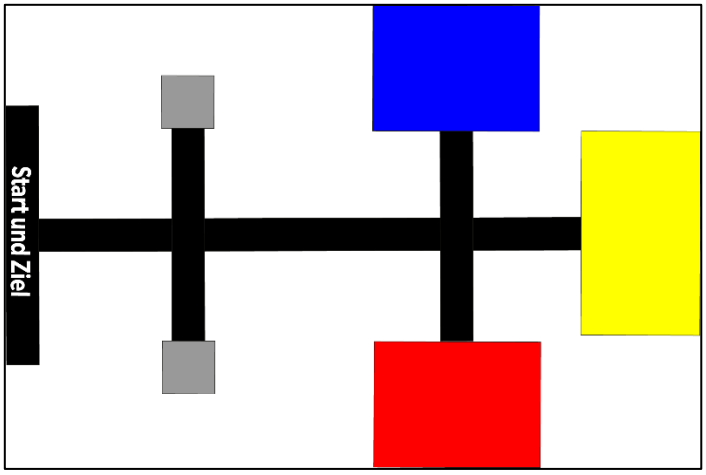


© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

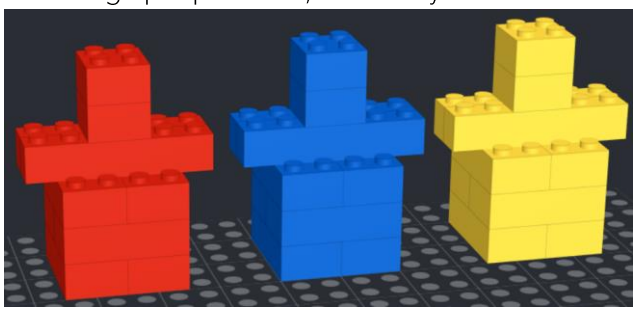
4 TRAINING TASK FOR PREPARING FOR A WRO COMPETITION

MATERIAL

- Your robot
- A playing field that looks like this:



- Three Lego people in red, blue and yellow:



INITIAL SETUP

Randomly select two of the three people and place them in the two gray squares. The third person will not be used. This means you have to write your code in such a way that your robot will work correctly no matter which two people you have.

Your robot always starts behind the Start/Goal line.

SUBTASK 1

Bring each person into the target area of the same color. Points will only be awarded for persons that are undamaged.

SUBTASK 2

Drive the robot back behind the Start/Goal line. Points are only awarded for this subtask if you have received some points for Subtask 1.

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

SCORING CHART

	Points	**Number**	**Total**
Subtask 1: Bring each person into the target area of the same color.			
The base of the person is completely inside the correct area.	20		
The person is partially inside the correct area.	15		
The person is completely or partially inside a wrong area.	5		
Teilaufgabe 2: Drive the robot back behind the Start/Goal line (only if points were given in Subtask 1).			
The robot has completely crossed the line.	20		
The robot has partially crossed the line.	10		
	Total points:		

IDEAS FOR SOLVING THIS TASK

> **Note**:
> There is no one single correct solution for this task. Many roads lead to Rome, and there are many ways to solve this problem. We will show one way that can work. But we will not give a complete solution. You will get a structure to work with, and you will then need to feed it with a lot of details. This task can serve as an inspiration to experiment yourself.
>
> In this chapter we use pseudo-code: It shows us the general structure (or scaffolding) of our code, but all details are missing.
>
> For our considerations we assume that the robot has one color sensor to detect the color of the persons, and one "hatch" to transport exactly one person at any given time. If we construct the robot so it can transport two people at once, of course our considerations and the structure of the code would look very differently.

We are going to focus on Subtask 1, which is quite complex. There are three possible objects, but only two are in pay, and we do not know which ones they are and which is on which start position. And they have to be taken into the area of the same color.

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

STRUCTURE OF THE CODE

We can break down the task into several smaller steps:

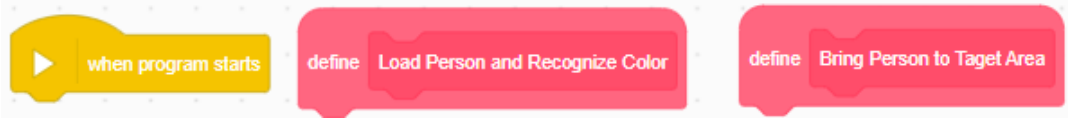
1. Drive from the start area to person area 1.
2. Load person and recognize color.
3. Drive to the correct target area and unload person.
4. Drive to person area 2.
5. Load person and recognize color.
6. Drive to correct target area and unload person.

Now think: Are there parts that are used more than once and that we can move to MyBlocks?

You will notice that steps 2 and 5 are identical, and steps 3 and 6 are identical.

So it will make sense to turn these into MyBlocks.

Let's start with this structure:



Under the main stack (**When program starts**) we will place everything that changes, under the other two stacks everything that stays the same.

STEP 1: DRIVE FROM THE START AREA TO PERSON AREA 1

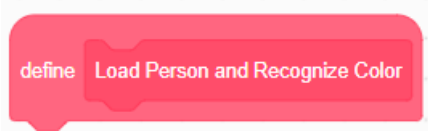
This is the pseudo-code for the first step from our list above: All details such as distances or how exactly the turn is done, are missing. You can worry about them later.

- Drive ahead to the first crossroads.
- Turn left.

This isn't very much! Maybe you are thinking now that you will still have to drive ahead until you get to person area 1. But look closely at the playing field: The distance from the first crossroads to Person Area 1 is exactly the same as the distance from the crossroads to Person Area 2! And we have learned that things that stay the same are put into a MyBlock (Remember the golden rule…). Therefore the "drive ahead until you get to the person" part will go into the MyBlock "**Load person and recognize color**".

STEP 2 AND 5: LOAD PERSON AND RECOGNIZE COLOR

Now let's see what goes into this MyBlock:



© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

- Drive ahead until you reach the person
- Open your hatch (Note: you may have to open the hatch earlier so you don't knock over your person! The space is limited, so you will have to text when the best moment is to open you hatch.)
- Load person (close loading mechanism).
- Read the color scan of the person and write the value into a variable called "**Person Color**".
- Drive back to crossroads.

We can now continue in the **main stack**.

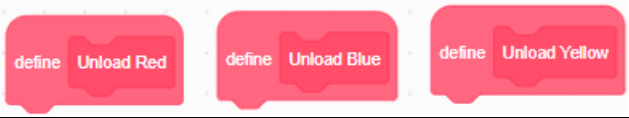
- Now we need to turn right so our robot faces the direction of the target areas. This bit is no longer in the MyBlock because after picking the second person up at Person Area 2, we have to turn the other way! Remember, only things that stay the same are put into MyBlocks!

STEPS 3 AND 6: DRIVE TO THE CORRECT TARGET AREA AND UNLOAD THE PERSON

Now we can continue with the other MyBlock, "**Bring person to target area**".

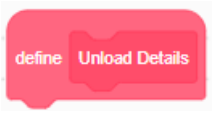
- Drive ahead until you get to the second crossroads.
- Now we need to read the value of the variable **Person Color** we wrote earlier.
- We have to decide: If **Person Color = red:**
    - Drive to red target area (turn right, then drive ahead)
    - Unload your person
    - Drive back to second crossroads and turn right
- Else
    - If **Person Color = blue:**
        - Drive to blue target area (turn left, then drive ahead).
        - Unload person.
        - Drive back to second crossroads and turn left.
    - Else
        - Drive to yellow target area (drive ahead)
        - Unload person
        - Drive back to second crossroads and turn 180 degrees.
- Drive to first crossroads.

You will notice that even the pseudo-code, which is leaving out many details, this MyBlock is getting somewhat long. Therefore we will create three new MyBlocks: **Unload Blue**, **Unload Red**, and **Unload Yellow**. The MyBlock **Bring Person to target area** will only contain the structure with the If-else-then blocks, all the rest with all the driving instructions will be placed in the specific Unload MyBlocks.



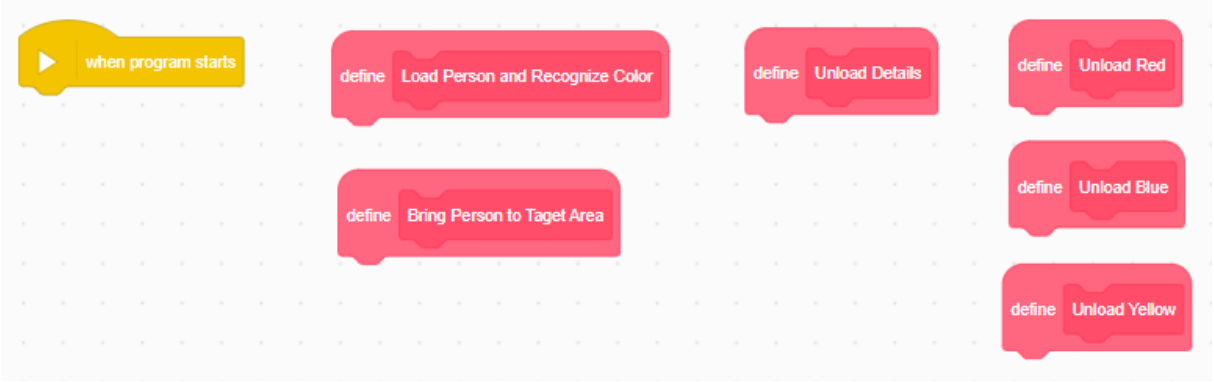
© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

And then we realize: The distance from the crossroads to the three target areas is always the same. Thus, the code we need for driving from the crossroads to the target area, unloading the person, and driving back to the crossroads, is always the same. So, we will place all of that into yet another MyBlock which we call **Unload Details**. The Unload [Color] MyBlocks will then contain only the turn from the crossroads into the correct direction, the MyBlock Unload Details, and then another turn at the crossroads.



You see: MyBlocks can be stacked inside each other as much as you like, just like the Russian nesting dolls!

So for the actual coding, this is our structure:



It is now up to you to fill each part with the necessary details!

Note:
The SPIKE APP uses numbers for each color. The most important ones are:

- white = 10
- black = 0
- red = 9
- yellow = 7
- blue = 3
- green = 5

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

Now you can write your code bit by bit and test each part individually.

SUBTASK 2

In the end, we only need to drive our robot back over the Start/Finish line. That's simple, one line of code is enough!

RESULT

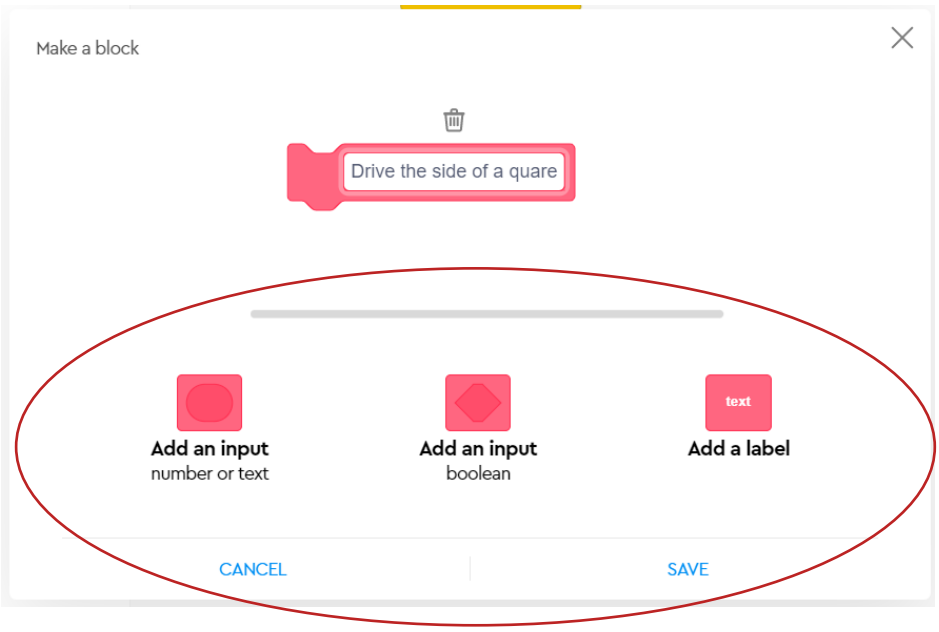
At the end of this dossier, you will find one possible solution for this task. Try to write the code without cheating though! Please note that the precise numbers in all the "move" blocks will depend on how you build your robot and thus are not set in stone. You will need to do a lot of testing to make sure the driving is correct, but the solution will show you the whole logic of the code.

And remember: There are many ways that lead to Rome, and there are many ways to solve this problem. Maybe you can think of better ways, or you build a robot that can transport two people at a time, and then your solution looks very different!

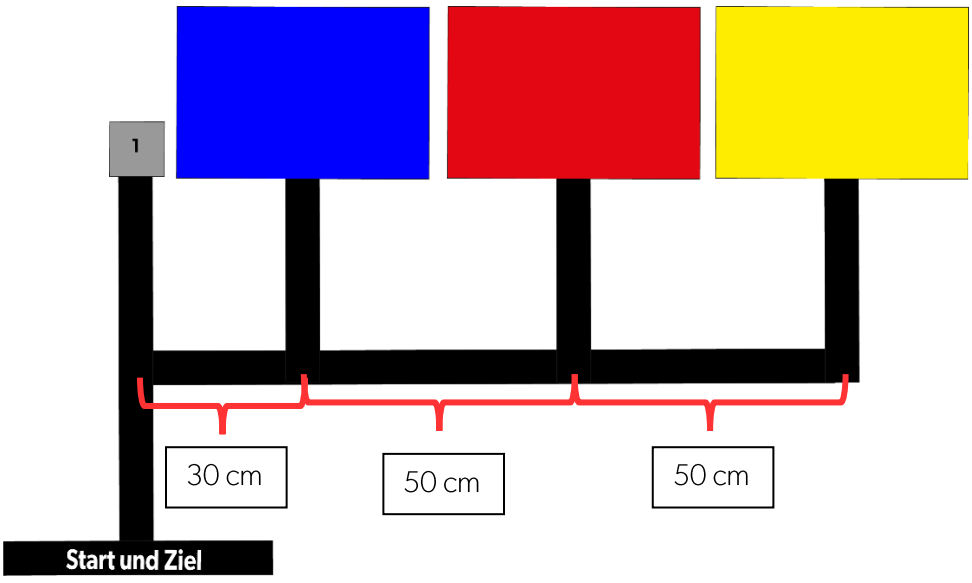
© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

5 USING MYBLOCKS WITH PARAMETERS

When we introduced MyBlocks in chapter 3, we skipped over the bottom part of the window where you name your MyBlock. This chapter will be dedicated to that part.



For this chapter we change the playing field of the WRO Training exercise a little:



This time, we have only one Person. It can be blue, red or yellow, and is randomly selected. And it has to be transported to the correct target area.

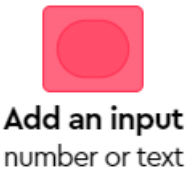
When you look at the playing field you will notice that the paths needed for dropping off our person will be very similar for all three options:

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

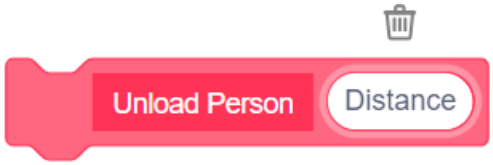
- Drive back to first crossroads.
- Turn right.
- Move ahead to (first/second/third) "target road" and turn left.
- Drive ahead to target area.
- Unload Person.

Now we need to find a way to make this possible with just a single MyBlock! And for that we will use parameters.

1. Let's create a new MyBlock called "Unload Person". Write the name of the MyBlock, but do not save yet!
2. And in this MyBlock we want to be able to have an option for each of the three different distances between the first crossroads and the Target Roads. And that's where a parameter comes in handy! This parameter will be a number, so we need the rounded parameter shape.
   → Click on the rounded parameter shape "Add an Input"

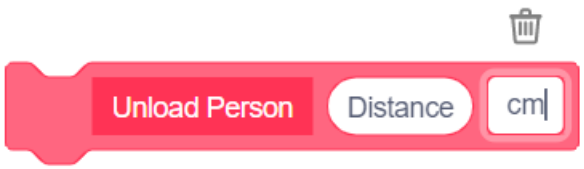


3. You will now see that the MyBlock gets a new input field, and we want to give that field a name. Let's call it "Distance".



4. Maybe we don't remember later what this parameter stands for, so we can give it a label. This is especially helpful if you add more than one parameter to your MyBlock! Click "Add Label".

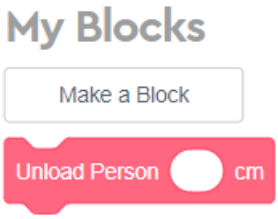
5. Since we have a parameter called "Distance" it makes sense to use the label "cm".



6. By the way, if you need to delete a parameter or label you can do so by clicking on the little trash icon above it.
7. Now you can save your MyBlock.

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

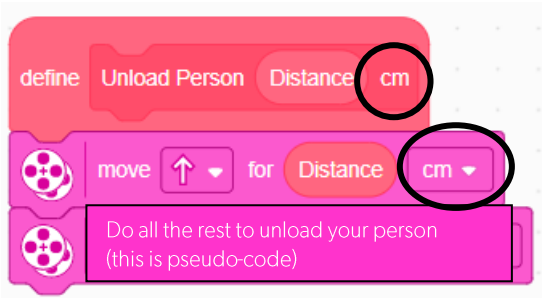
8. Your MyBlock will now look like this in your list of blocks:



9. And the MyBlock Head will look like this:

Now how does that help us? Go back to the image with the new playing field. You will see that the distance between the various crossroads is given in centimeters. Does that ring a bell? That's exactly what our parameter is used for!

So let's write the code for the MyBlock:

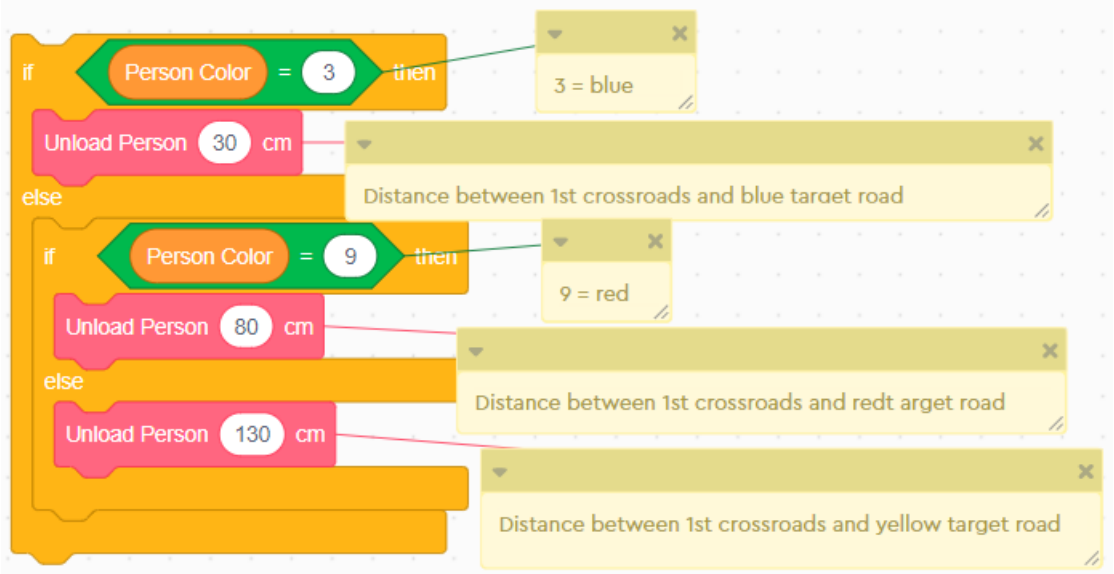


10. Drag the **Distance** parameter into the Input field of the **move** block.
11. Make sure the unit in the move block is set to **cm**, just like the label in the MyBlock Head!
12. Write all the rest of the code you need for dropping off your Person.

Now let's go to the Main Stack. To save time, we start after the Person has been loaded and scanned and the color of the person written into the Variable "**Person Color**". We did all that in the last chapter, no need to repeat it here.

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

Here is the code snippet:



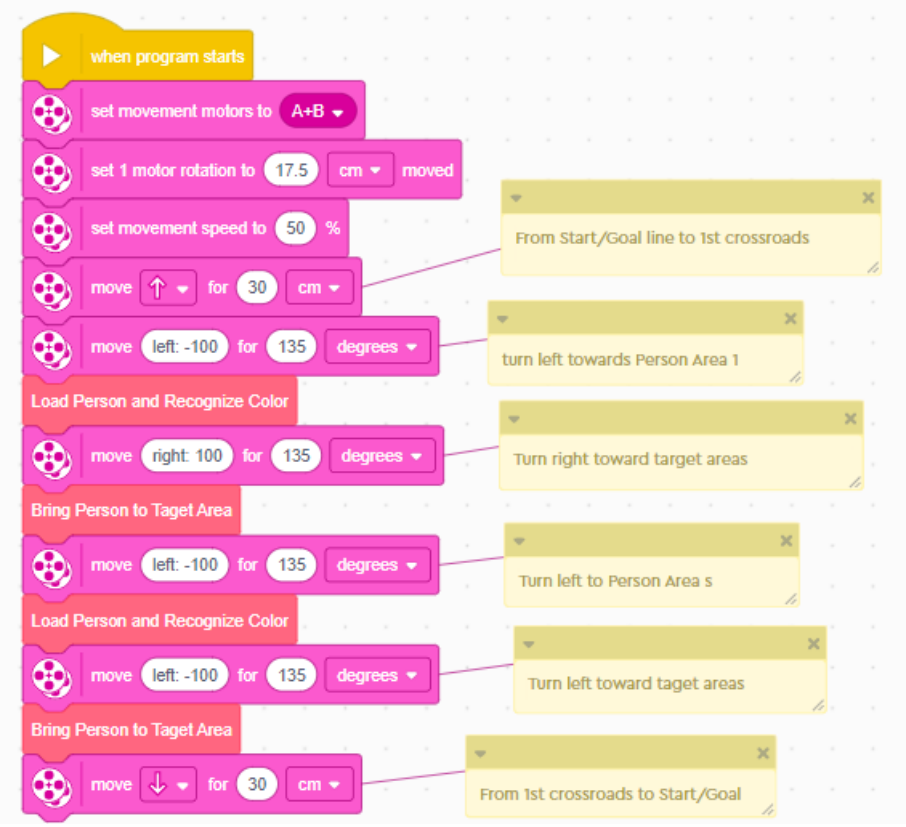
SOLUTION TO WRO TRAINING TASK IN CHAPTER

RESULT MAIN STACK:

If we've done everything right, this is what our main stack will look like[4]:

[4] Please note that the precise numbers for turns and distances are estimated and not tested. Depending on your robot build, you will need different numbers here!

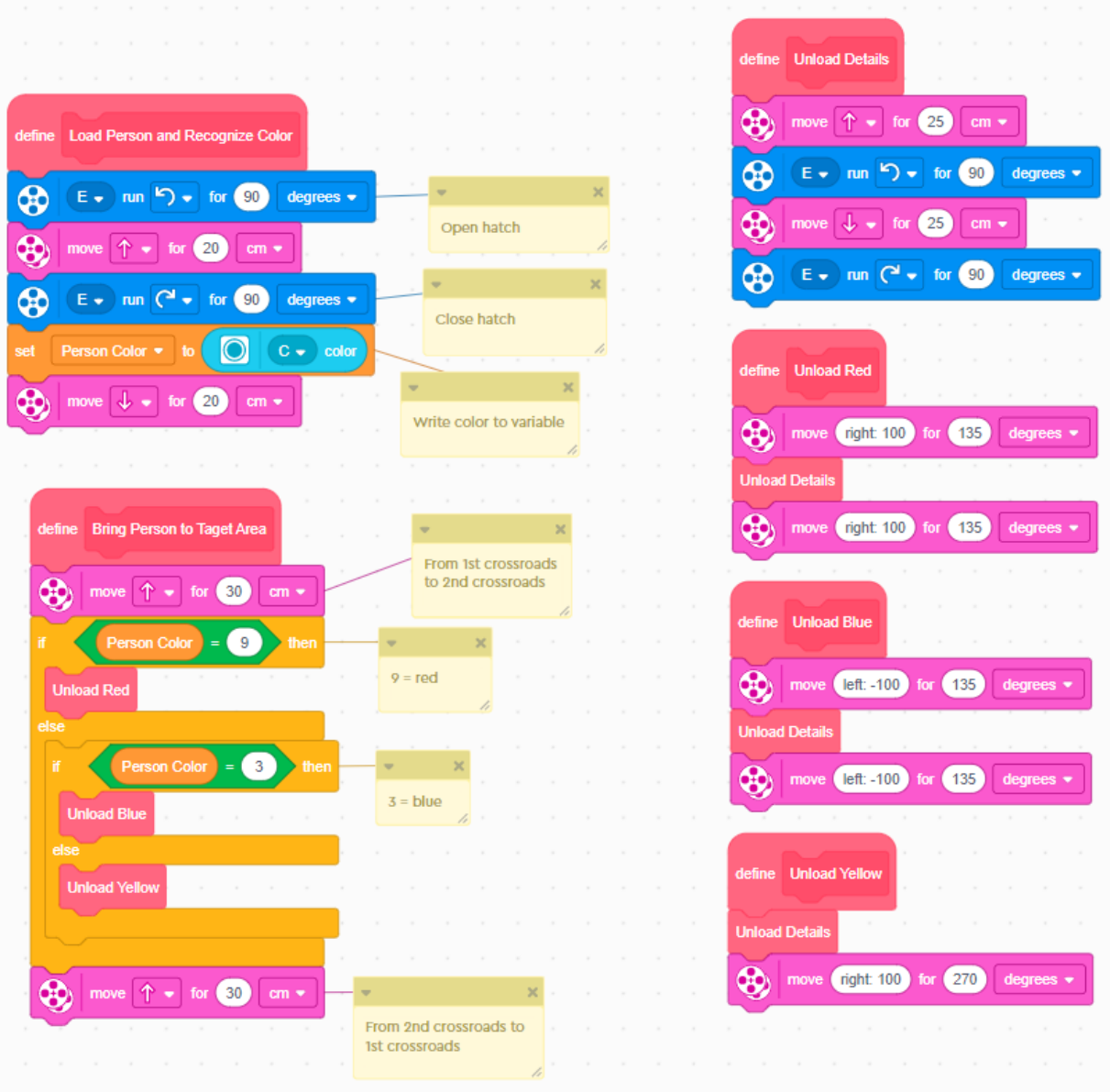
© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.



These are 14 text blocks, including the 4 blocks in the header part. In addition to that, we have 6 very short MyBlocks. We might have to do some scrolling on our screen to make them all fit, or zoom out a little, but in the end it's very neat and easy to understand.

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.

RESULT MYBLOCKS



Now if we tried to do all of that in just one single main stack without the MyBlocks, our code would be much, much longer. I tried it just to see what it looks like, and I had to write almost 100 lines of code. And this is still a fairly simple task, and only one part of a WRO challenge. Such long stacks are sometimes called “toilet paper code», and it’s easy to see why…

© 2022 Verein World Robot Olympiad Schweiz – Offizieller Organisator der World Robot Olympiad in der Schweiz
WRO® und World Robot Olympiad™ sind Markenzeichen der World Robot Olympiad Association.