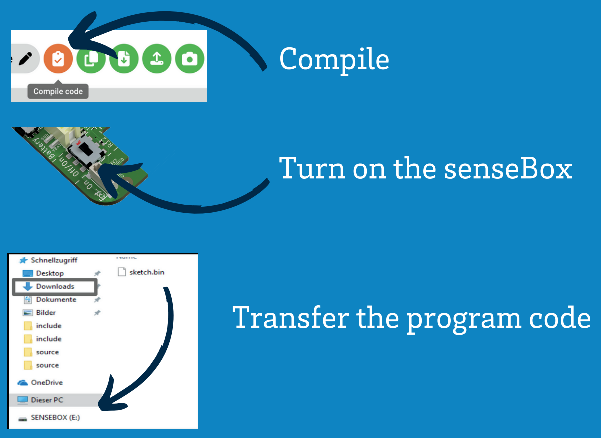
Info: Uploading the program code


Compile

Turn on the senseBox

Transfer the program code

Alternative: senseBox Connect App
Instructions for transferring the program code using a tablet:

senseBox

Car Traffic Counter

Blockly | Level: ★★★ | 30 min.

Cities are often full of cars that produce emissions and noise. This is why discussions about car-free city centers keep coming up. But how many cars are actually in the city? In this project, you will build a traffic counter and find out!

The code isn't working? Troubleshooting tips

- Are your cables plugged in exactly as shown in the illustration?
- Are your command blocks really connected like small "puzzle pieces"?
- Have you deleted all blocks that are not connected to your main block?
- Have you compiled the latest version of your program code and uploaded it again after making changes in Blockly?

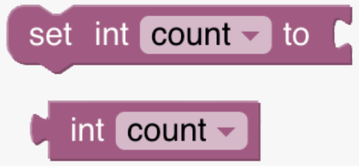
Still having trouble?
Then get in touch with a mentor!

iCODE Variable

INFO: VARIABLES

When programming the traffic counter, you will repeatedly need to access certain values. To make this easy, computer science offers a helpful solution with the concept of variables. They are like a box with a name on it – you can store different things inside this box, such as numbers or text, and retrieve them later.

Depending on what you want to store in a variable, you need to choose the correct data type:



```
set int count to
int count
```

Character (char): For single text characters
Text (string): For whole words or sentences
Number (int): For numbers from –32,768 to +32,768

Large number (long): For numbers from –2,147,483,648 to +2,147,483,648
Decimal number (float): For numbers with decimal places (e.g., 25.3)
State (boolean): true or false

Variables can change their value during the program. For example, you can increase the variable "count" by one each time a car passes by.

<iCODE>MS

Kofinanziert von der Europäischen Union

Ministerium für Wirtschaft, Industrie, Klimaschutz und Energie des Landes Nordrhein-Westfalen

senseBox

Car Traffic Counter

Connect the OLED display and the sensors to the microcontroller.

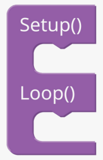
1) Create a program so that the measurement values from the distance sensor are shown on the display.

2) Extend your program code so that the number of detected cars is also shown on the display.

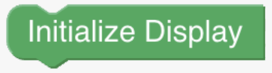
Step 1A

1. For programming: blockly.sensebox.de

2. In Setup, some components need to be activated once at the beginning of the program.

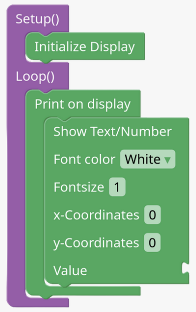

Setup()
Loop()

3. The display has to be initialized in the setup:


Initialize Display

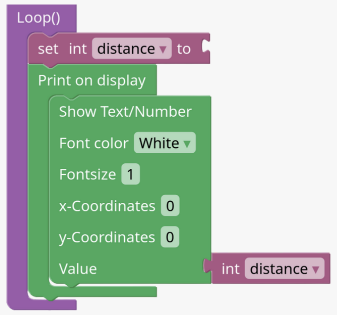
Step 1B

1. To display the measurements on the screen, you need the blocks "Print on display" and "Show Text/Number" in the loop.

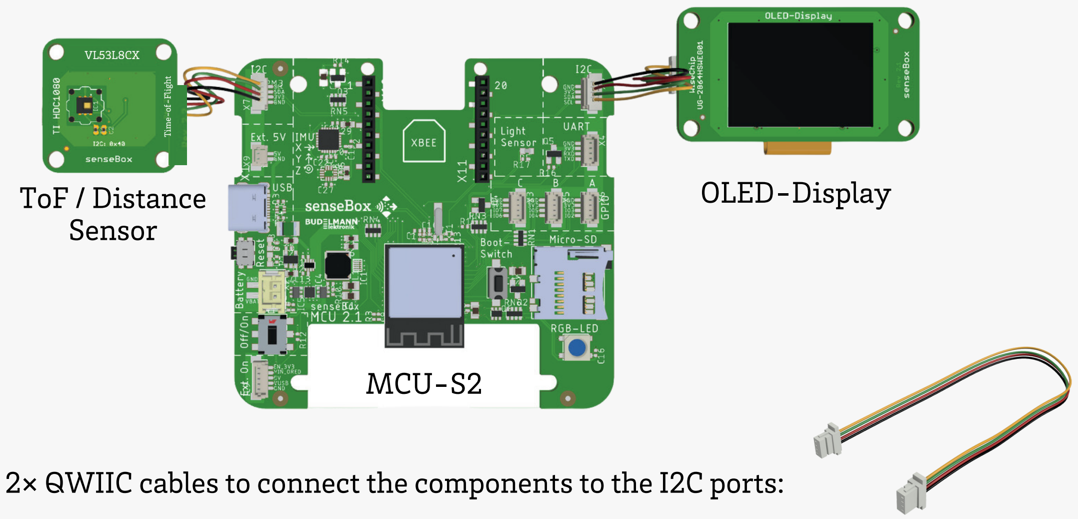

Setup()
Initialize Display
Loop()
Print on display
Show Text/Number
Font color White
Fontsize 1
x-Coordinates 0
y-Coordinates 0
Value

2. Instead of inserting the sensor block directly into "Value", you define a variable and then insert this variable into "Value". To do this, select "Create Typed Variable" in the Variables category.

3. Define the variable as a number (int), name it "distance", and insert it into the loop.


Loop()
set int distance to
Print on display
Show Text/Number
Font color White
Fontsize 1
x-Coordinates 0
y-Coordinates 0
Value int distance

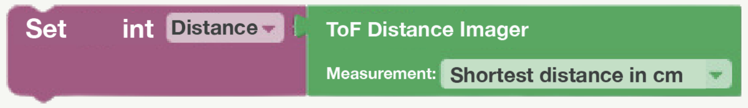
Hardware-Setup



2× QWIIC cables to connect the components to the I2C ports:

Step 1C

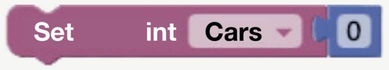
The variable now has to be assigned a measurement value. Since the distance is measured with the ToF sensor, this sensor block is connected to the variable "distance".


Set int Distance ToF Distance Imager
Measurement: Shortest distance in cm

Test your code!

Step 2

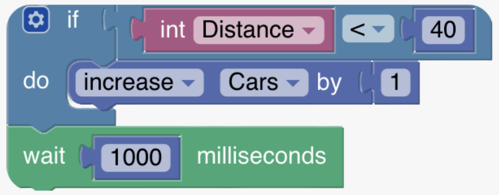
1. The measuring device now detects distances, but not cars yet. To count them, you need to create a new variable (int) called 'car'. Since the car count should start at zero, you set this in the 'Setup' section (category Math).


Set int Cars 0

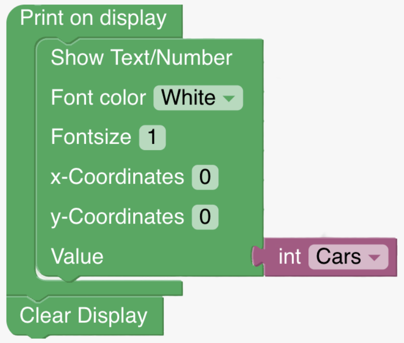
2. With an "if – do" condition, you define when a new car should be counted:

If the distance is less than 40 cm, then the car variable increases by one.

To prevent a car from being counted twice, add the block "wait 1000 milliseconds".


if int Distance < 40
do increase Cars by 1
wait 1000 milliseconds

3. To display the number of cars on the screen, add the "Show Text/Number" block again and insert the variable "cars" as the value. By regularly clearing the display, you ensure that the most up-to-date and correct distance to the passing car is shown. Also adjust the y-coordinate so that the distance and the count appear one below the other.


Print on display
Show Text/Number
Font color White
Fontsize 1
x-Coordinates 0
y-Coordinates 0
Value int Cars
Clear Display

This completes the programming. Now simulate passing cars using objects and test whether everything works.

Test your code