Co-funded by the European Union

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.

Programming the DIY robotic car to follow the instructions received by the created application

Introducing the 5 Big Ideas in Artificial Intelligence using Internet of Things in STEM education

T2.4 IoT Projects Design & Resources Development

13.10.2023 | EDUMOTIVA
PROJECT NUMBER: 2022-1-FR01-KA220-SCH-000085611

AI4STEM IoT Projects Design & Resources Development
Project: The DIY robotic car
Programming the DIY robotic car to follow the instructions received by the created application

Copyright

Co-funded by the European Union

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.

Table of Contents

Co-funded by the European Union

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.

1.1 Introduction

The following image (Figure 1) presents the entire script that should be uploaded to micro:bit in order to enable our robotic car to follow the instructions received by the application. Concerning the needs of this project, this part is not mandatory to be taught in depth. Parts that are included in this script, such as the way that the motors can be activated and programmed to move to different directions (i.e., forward, backwards etc.) are explained in the "T2.4_WarmUp_programming_activities_for_the_robotic_car.pdf" file. Therefore, and depending on your students' level, you can either briefly explain the key points of the script, or you can instruct them to download the relevant .hex file to the micro:bit.

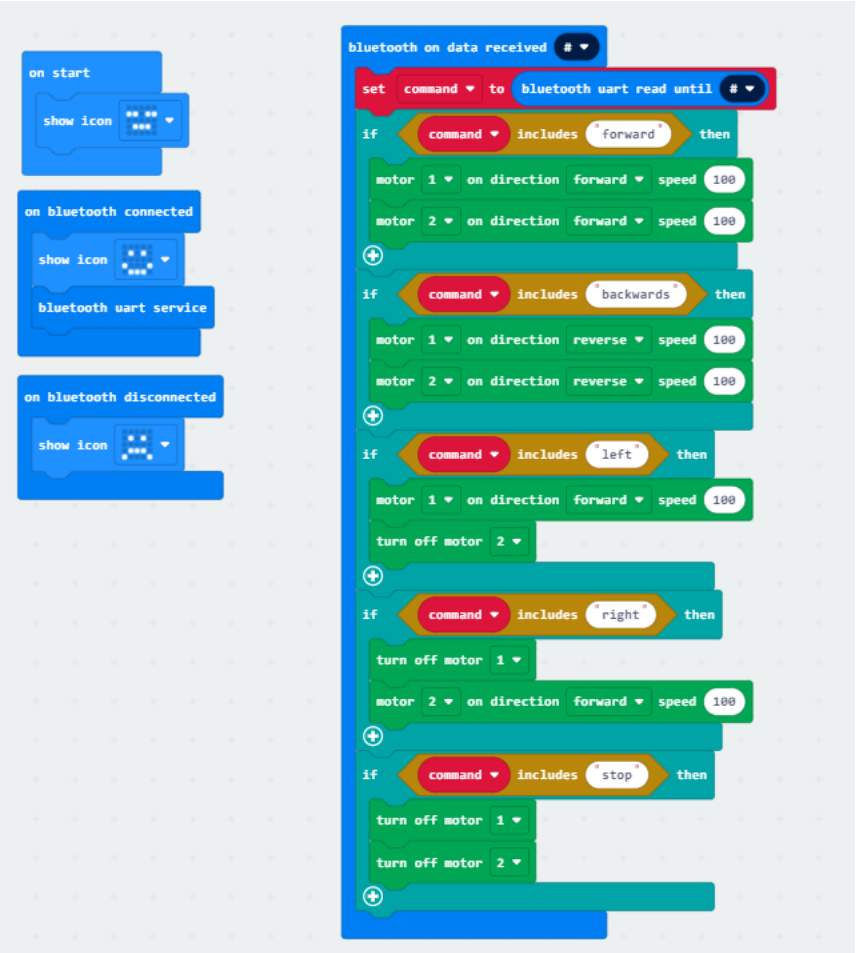

Figure 1: The entire script

The above script can be roughly divided in two sections, namely the "Setup & Connectivity" section and "Actions Taken after Receiving Bluetooth Message" section (*Figure 2, Figure 3*).

Co-funded by the European Union

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.

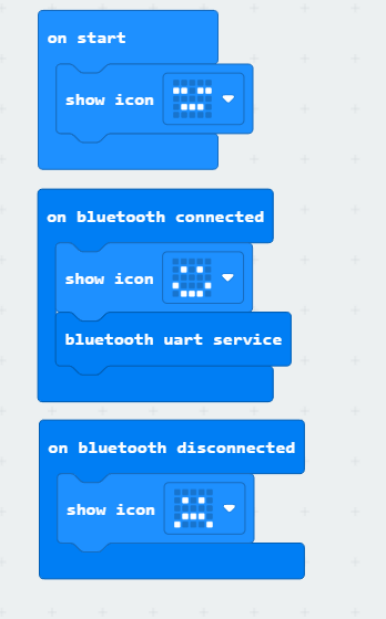

Figure 2: Setup & Connectivity

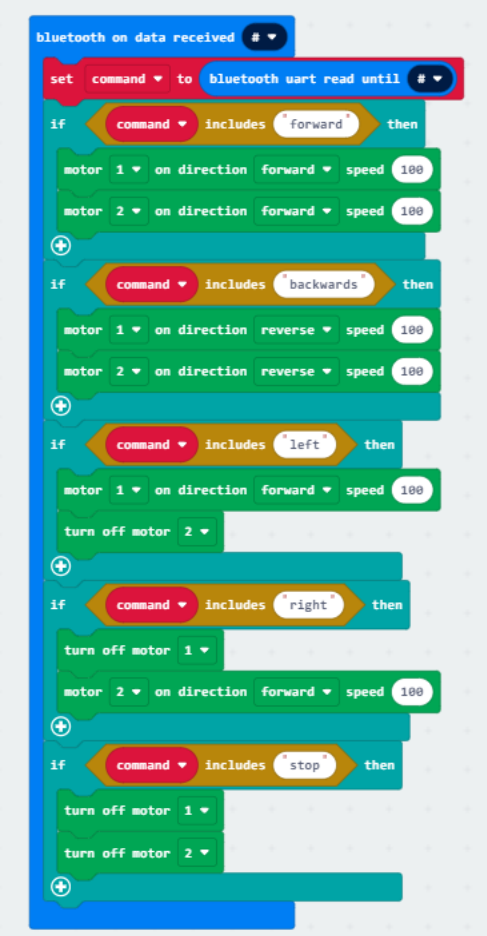

Figure 3: Actions Taken after Receiving Bluetooth Message

Co-funded by the European Union

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.

1.2 Adding the Bluetooth extension

The first step towards creating the script is to add micro:bit's Bluetooth extension to MakeCode, in order to enable this feature for programming purposes. This procedure is the same as the one described on page 5 of the document "*T2.4_WarmUp_programming_activities_for_the_robotic_car.pdf*", with the difference that we search for "Bluetooth" in the extensions' search bar (*Figure 4*).

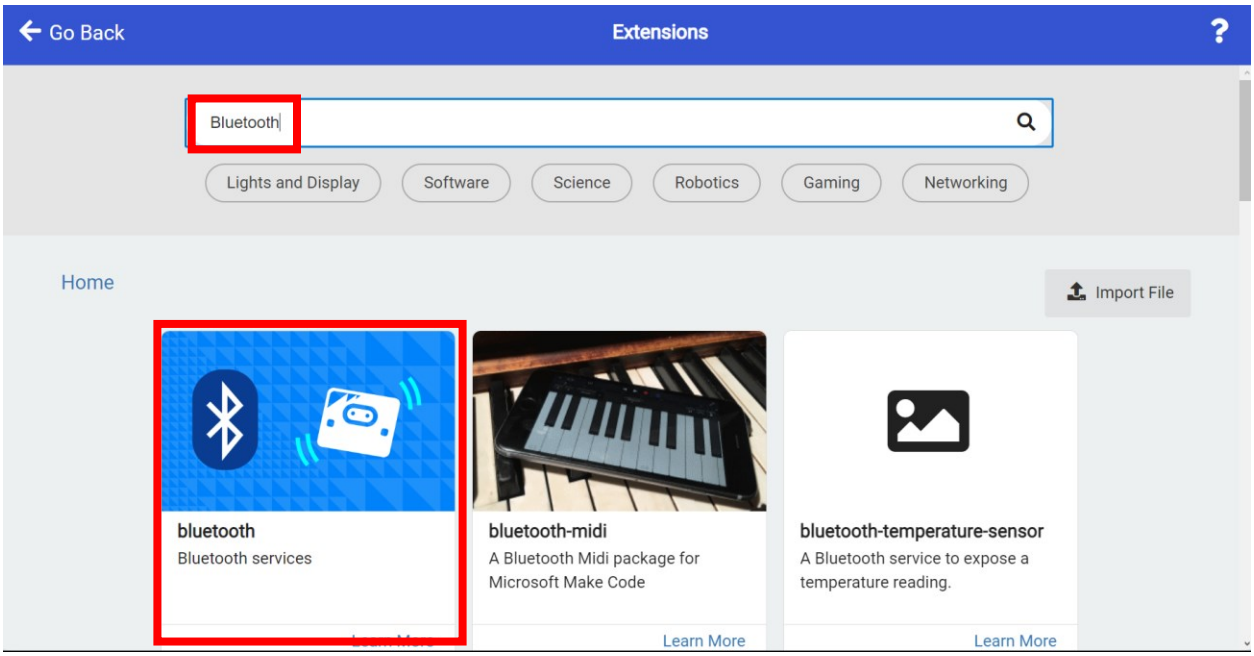

Figure 4: Finding the Bluetooth extension

After adding the extension, a new menu with block of commands can be found in the MakeCode environment (*Figure 5*).

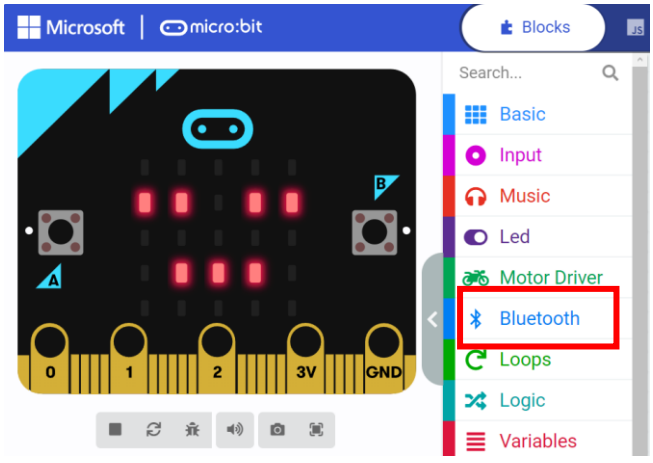

Figure 5: The Bluetooth menu.

Co-funded by the European Union

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.

Note: Before adding the new menu, a warning/notification will pop-up (Figure 6), informing that the radio extension is incompatible to Bluetooth and should be removed. Click on the "**Remove extension(s) and add bluetooth**" button to confirm your choice.

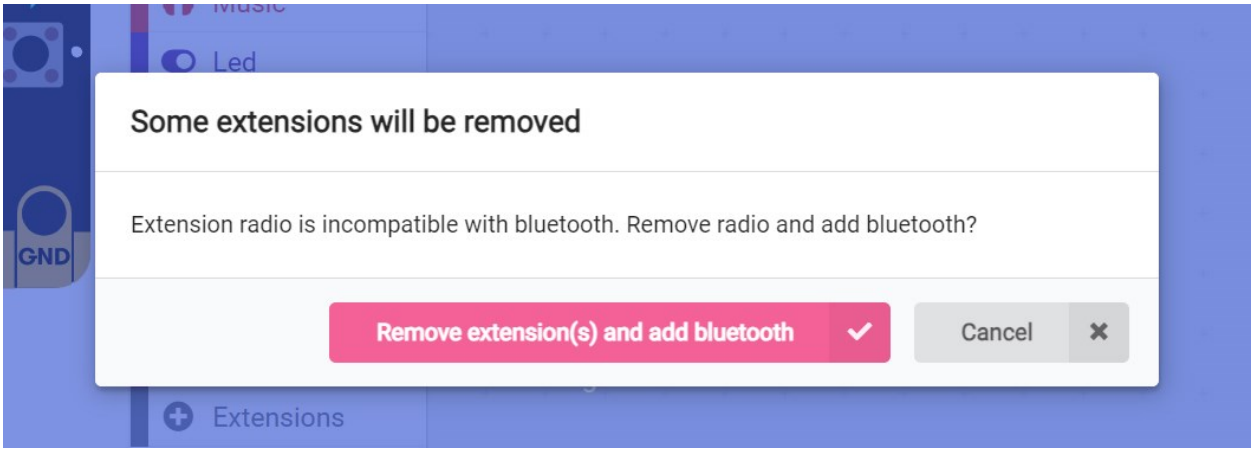

Figure 6: Warning regarding extensions that will be removed

1.3 Creating the "Setup & Connectivity" section

This section (*Figure 7*) consists of three main blocks: a **Basic** block called "**on start**" and two **Bluetooth** blocks called "**on bluetooth connected**" and "**on bluetooth disconnected**". The first specifies what happens when the Micro:bit turns on, the second one what happens after the Bluetooth connection is established and the third one the actions taken when the Bluetooth connection is terminated.

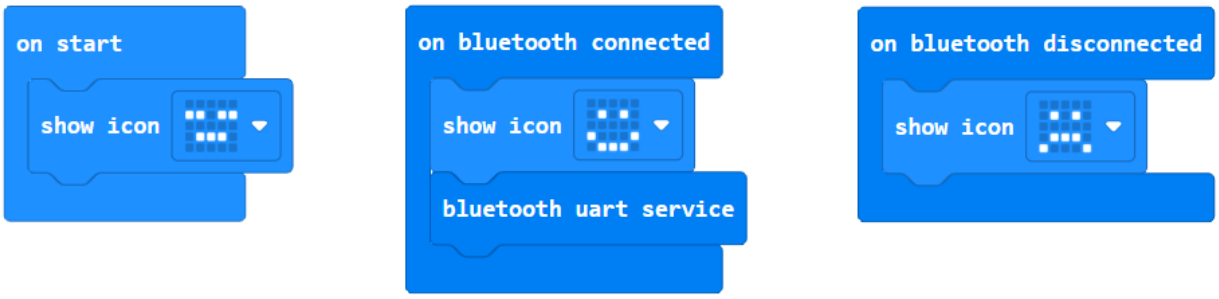

Figure 7: The "Setup & Connectivity" section

In detail:

- **On start** (when the micro:bit turns on), **show** an asleep **icon**. This icon will be displayed on micro:bit's LED screen and is used to check that the script has been successfully downloaded to the board. You can either choose the asleep icon or anything else you like from the floating menu.

Co-funded by the European Union

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.

- **On bluetooth connected** (when a Bluetooth connection has been established), **show** a “happy face” **icon** and start the **bluetooth uart service**. Through the bluetooth uart service the micro:bit is enabled to send and receive messages via Bluetooth. Furthermore, the "happy face" icon is used as an indicator of a **successful** Bluetooth connection.

- **On bluetooth disconnected** (when the Bluetooth connection has been terminated), **show** a “sad face” **icon**. The "sad face" icon is used as an indicator of a **terminated** Bluetooth connection.

After completing all of the aforementioned steps, your micro:bit will inform you about the connectivity status of any Bluetooth service, and will be able to receive messages from other Bluetooth devices.

1.4 Creating the “Actions Taken after Receiving Bluetooth Message” section

This section describes what will happen when a message is received via Bluetooth. In order to enable micro:bit to recognize a message, the message should be placed between 2 hashtag (#) symbols (e.g. #message#). Then, the message is communicated, and based on its content, the corresponding command is executed. For instance, and as depicted in the following table, if the received message contains the word forward, placed between 2 hashtags (i.e., #forward#) then, the robotic car will be instructed to move forward.

Message Received	Actual Message	If the message contains the word:	Then, execute the following action:
#message#	message	forward	move car forward
		backwards	move car backwards
		left	move car left
		right	move car right
		stop	stop the car
		none of the above	-

Co-funded by the European Union

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.

Figure 8, presents the part of the script that allows the aforementioned communication.

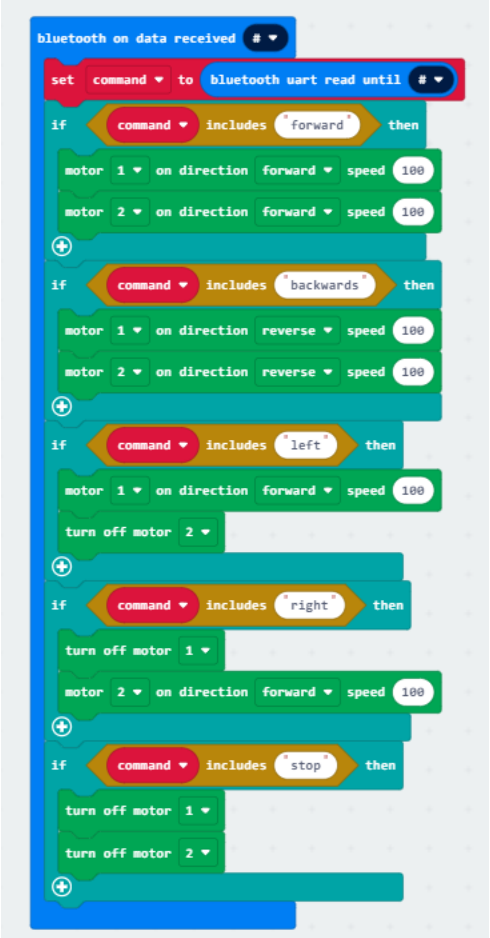

Figure 8

In detail:

On bluetooth data received (when a Bluetooth message is received), **read** it through **bluetooth uart until** you see **#** and then **set** it as a variable named **command**. Next, do the following checks:

- **If command includes** the word “forward”**, then turn on** both **motors** on the **forward direction** with a **speed of 100**
- **If command includes** the word “backwards”**, then turn on** both **motors** on the **reverse direction** with a **speed of 100**
- **If command includes** the word “left”**, then turn on** the **1**st **motor** on the **forward direction** with a **speed of 100**, and **turn off** the **2**nd **motor**

Co-funded by the European Union

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.

- If **command** **includes** the word “right”, then **turn on** the **2**nd **motor** on the **forward direction** with a **speed of 100**, and **turn off** the **1**st **motor**

- If **command** **includes** the word “stop”, then **turn off** both **motors**.

Note: to create the variable “command” click on the “Variables”, and then on the “Make a Variable” button (*Figure 9*). On the pop-up menu, type the name “command” to the “new variable name” section, and press Ok. The “**set command to …** ” block of command will appear to Variables sub-menu.

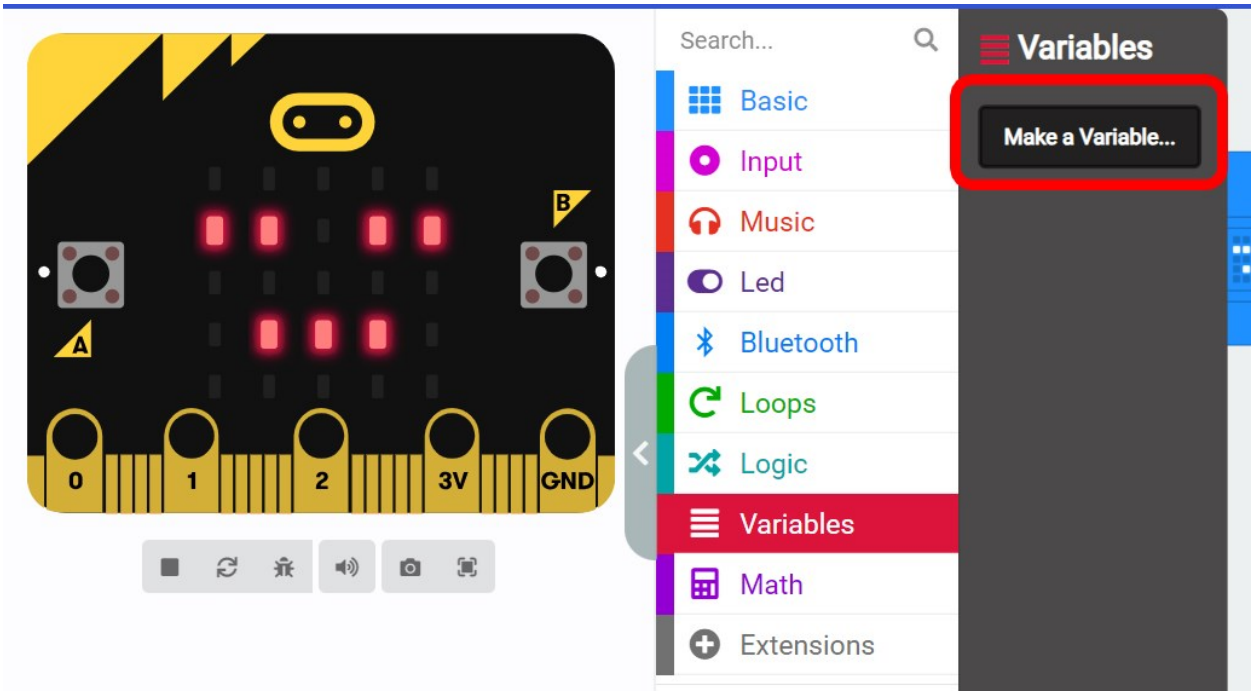

Figure 9: Creating a Variable

1.5 Adding indicators for received commands

This step is not mandatory, but it can help students realize whether or not the downloaded script is smoothly running.

In the “If…then” statement, add a “show leds” command, and check the corresponding boxes, in order to create an arrow showing the direction that the robotic car should follow, if the respective command is received (e.g., create an up arrow if the received command is “forward”)

Co-funded by the European Union

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.

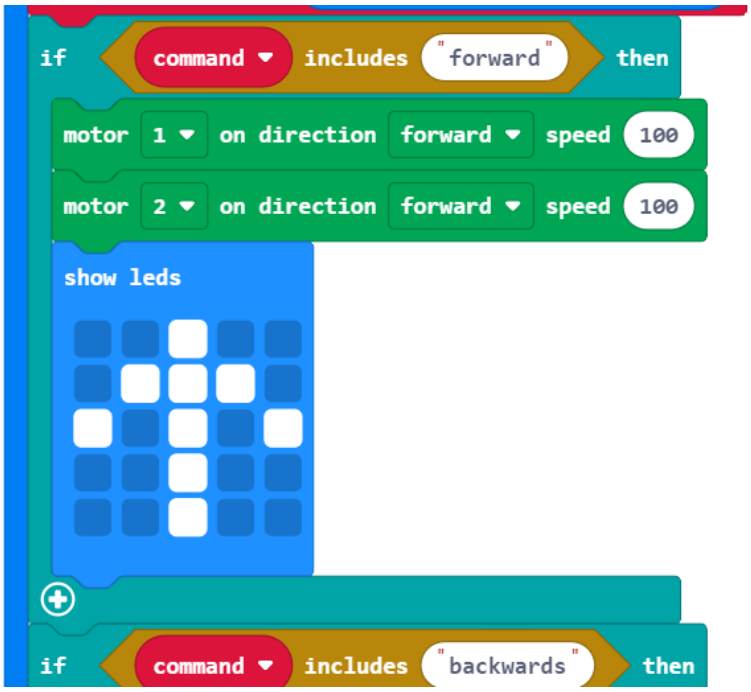

Figure 10: Adding a "show leds" command depicting a "front" arrow, as verification that the micro:bit has received the correct command corresponding to forward movement (similarly for the remaining commands)

Adding the "show leds" command in MakeCode to display a specific icon when a Bluetooth message is received on the micro:bit is a valuable enhancement to the project. This block serves as a visual indicator, making it easier to confirm that the micro:bit has successfully received and interpreted the Bluetooth command. Visual signals are particularly helpful in debugging and ensuring the proper functioning of the script in real-time.

Note: The "show leds" block is located within the Basic category.

Tip: Customize the LED pattern to represent your desired icon or symbol by turning on or off individual LED lights in the grid.

By implementing this step, not only the user experience is improved, but also the troubleshooting process is simplified, as immediate feedback on command reception is provided. Moreover, the project becomes more user-friendly and visually engaging.

1.6 Important notes regarding Bluetooth connection

A) Before downloading the final script to micro:bit, make sure that "No Pairing Required: Anyone can connect via Bluetooth" option is selected, in the Project settings menu. To check this parameter, click on the Gear symbol, and from the floating menu, click on the "Project settings" (*Figure 11*). Then activate the ""No Pairing Required: Anyone can connect via Bluetooth" option.

Co-funded by the European Union

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.

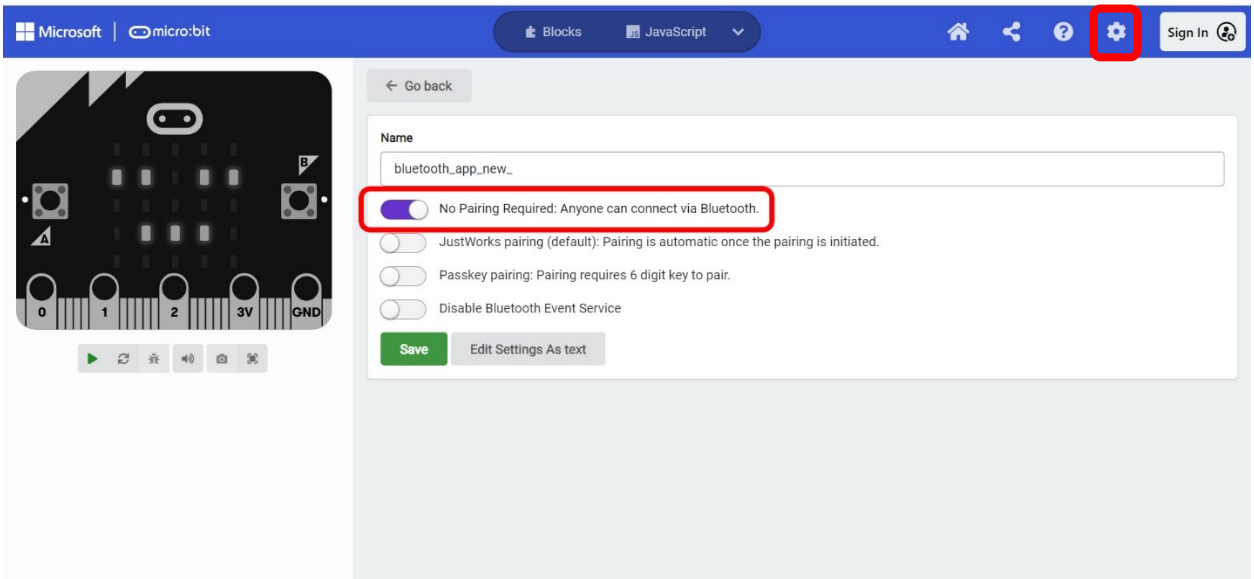

Figure 11: Checking the Project settings menu

B) Before trying to connect micro:bit to the created application make sure that your smart device, does actually recognize micro:bit. You can check this aspect, by opening the Bluetooth menu on your smart device, and checking whether or not the micro:bit board appear on the available connections.

If you cannot find your micro:bit to the available connections, try to rename your project by adding the word “microbit” to the beginning of the name (e.g. “microbit-robotic_car”), and download again the project to your micro:bit.

Co-funded by the European Union

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.