Worksheet for students Team:.....

Time for brainstorming:

What do you know about the voice assistant applications embedded in cars? Do you know if AI technologies have been used to optimize these applications? *Search with your team for information online and write your answers below.*

Can you think of some real-world cases where the voice commands for controlling and/or navigating an autonomous car could be useful or necessary? *Discuss with your team and document your thoughts below.*

Can you think of some advantages and disadvantages of using voice commands for controlling and/or navigating an autonomous car? How can these affect the design process? *Discuss with your team and document your thoughts below.*

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.

Time for creating the application for controlling/navigating our robotic car through voice commands

Eriching our App by integrating AI services

In order to navigate our robotic car by using our voice, we will add an extra button to our application. When this button is pressed, our device's microphone will be triggered and, with the help of Google's AI Speech-to-Text service, our voice commands will be converted to text. Then, this text will be filtered as follows: If the voice command contains the word "forward", "backwards", "left", "right" or "stop", then our application will transmit (through the Bluetooth) the corresponding message to our robotic car, and our robotic car will perform accordingly.

Try to fill the following table based on the aforementioned description. Which message will be sent to our robotic car, and which movement will be consequently performed, when each one of the voice commands will be given? Tip: You can also take a look at the Makecode script.

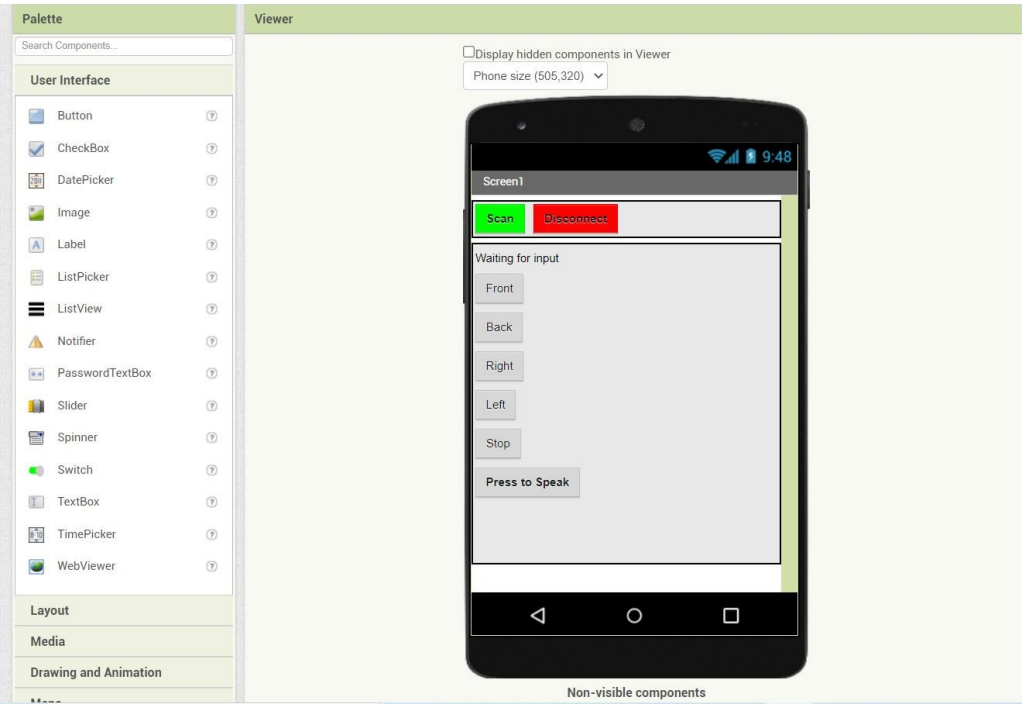
If the voice command contains the word:	*Then, send to the robotic car the message:*	*Then, the robotic car will:*
"forward"		
"backwards"		
"left"		
"right"		
"stop"		

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) Adding the new button

Go to the Designer menu and drag a Button from the User Interface tab to the design area of the interface of the application. Then, change the Button's text and the name of the Button to something meaningful such as "Press to speak" and "speak" in respect. After this step, the preview of your application will look like the following image.

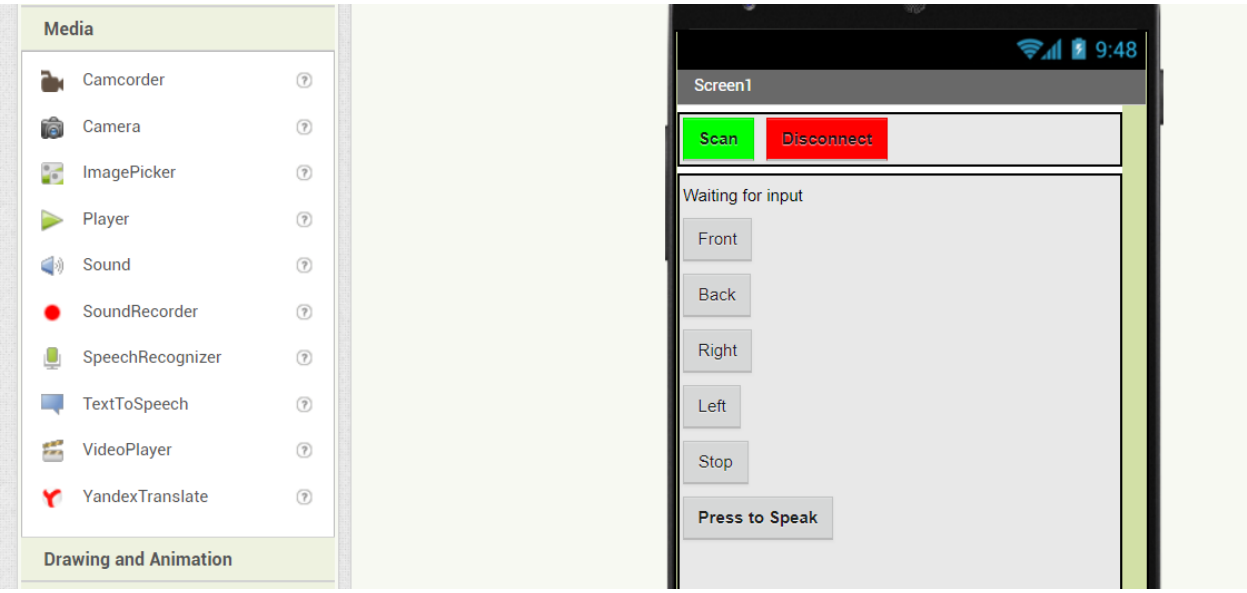


Co-funded by the European Union

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.

2) Adding the Speech-to-Text AI Service

Now, it's time to add a service that will enable our application to recognize our voice commands. Click on the Media tab. Which one of the contained services do you think we need to use? *Type your answer below.*



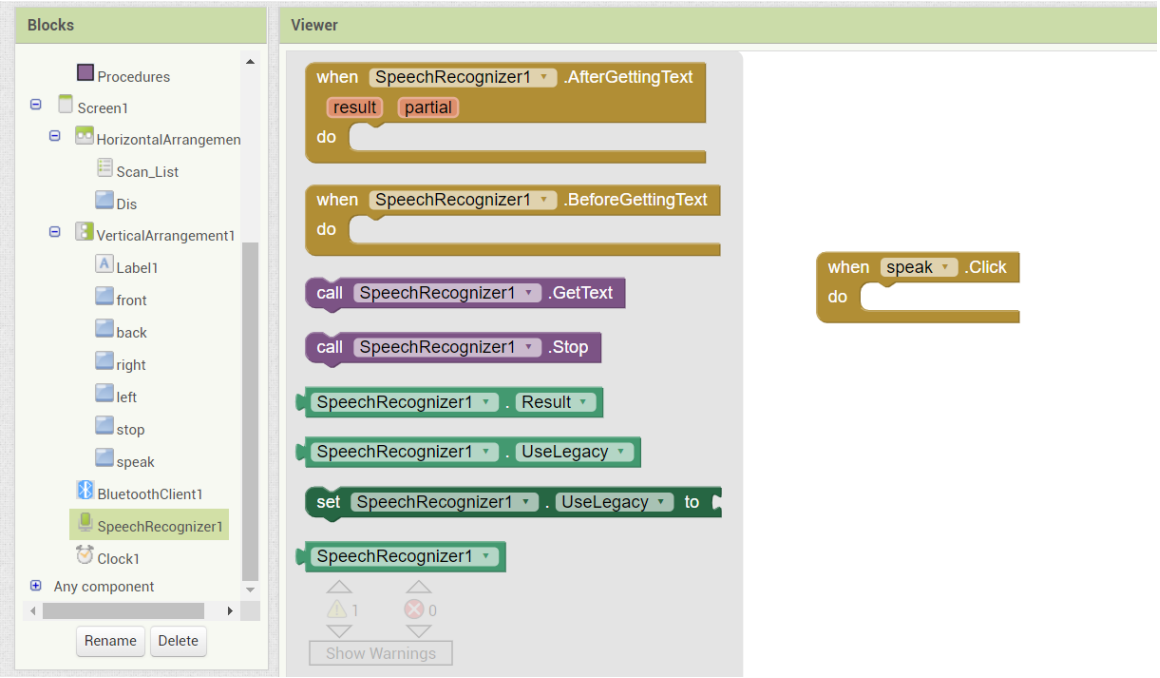
Next, drag this service to the application's screen.

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.

3) Creating the script for the "Press to speak" button

As mentioned earlier, when the speak button is pressed, our voice commands will be converted to text. From the commands contained in the following image, w*hich SpeechRecogniser command should we place inside the "when_speak_click" event handler, to enable our application to* ***get*** *a piece of* ***text****? Discuss with your team and type your thoughts below.*

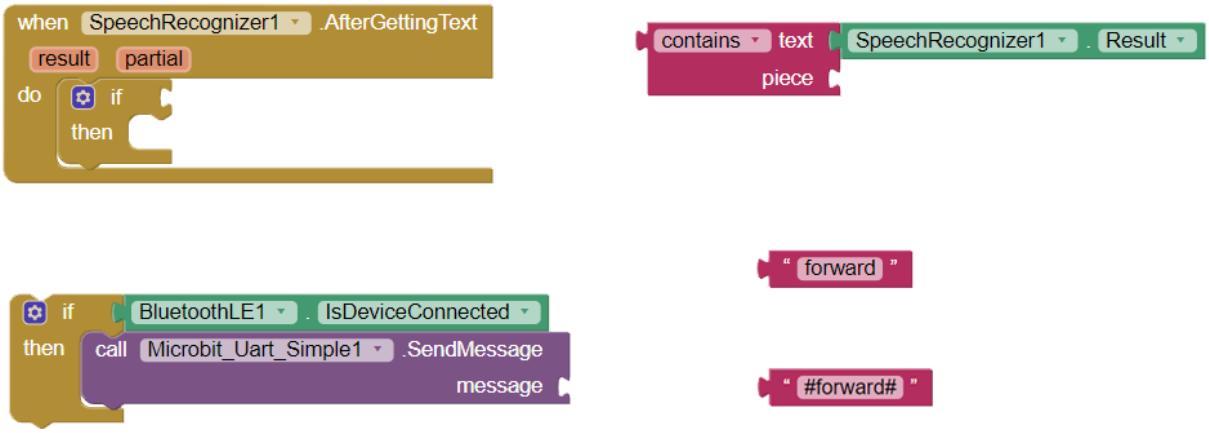


Funded by the European Union. Views and opinions expressed are however those of the author(s) only and do not necessarily reflect those of the European Union or the European Education and Culture Executive Agency (EACEA). Neither the European Union nor EACEA can be held responsible for them.

4) Filtering the received text

After programming what will happen when the Speak button is clicked (namely getting the voice commands as text)? it's time to program our application to recognize specific pieces of the received text in order to enable our robotic car to perform the corresponding movements.

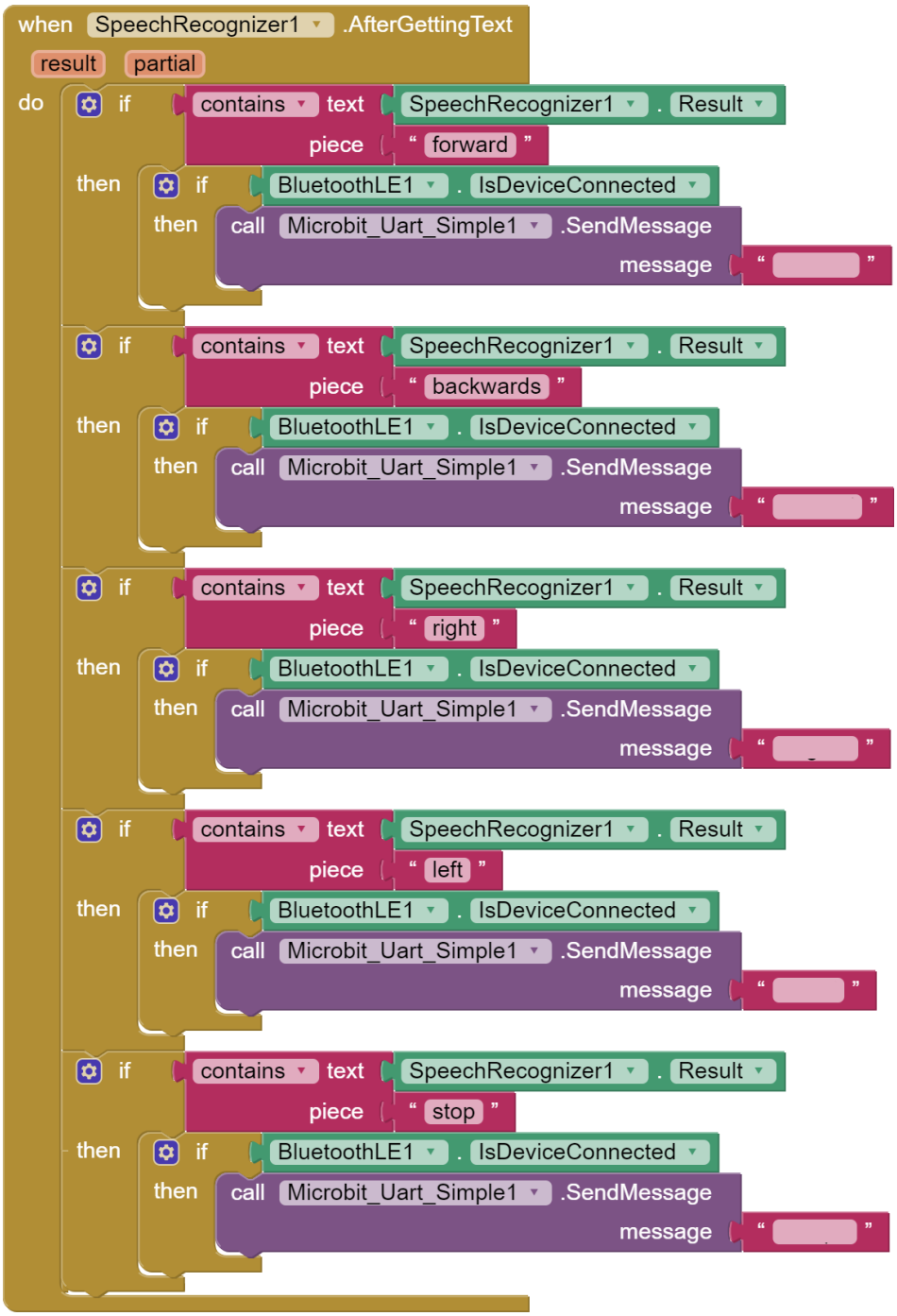
The following script is semi-structured. Try to place the commands in the correct order to program the application to move forward the robotic car (by sending via the Bluetooth the corresponding message) if the piece "forward" is contained to the recognized text.



Now fill the previous script with all the needed commands to enable your application to move the robotic car accordingly, if the corresponding voice command is recognized (i.e., forward, backwards, right, left and stop). To do that try to fill the following semi-structured script.

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.



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.

5) Test the application

Now, it's time to test your application. Go to the Build menu and generate your application. After installing the .apk file to your smart device, open it and test if your robotic car corresponds to the programmed voice commands, when you click on the Press to speak button.

Try to use different voice commands and check whether your robotic car responds to them or not. Note down your observations to the following table *(e.g., Voice command "Move forward" is successful, voice command "go to front" failed etc.).*

Voice command	Success	Failure

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.

Let's make our application less "boring" by adding a Clock sensor

So far, we have implemented the following scenario: Each time we want to give a voice command, we have to press the "Press to speak" button.

This solution is functionable, but it might not be very convenient, especially if we project this scenario to real life and real driving conditions (i.e., driving a car in a city).

To create an optimal solution we will modify our application as follows:

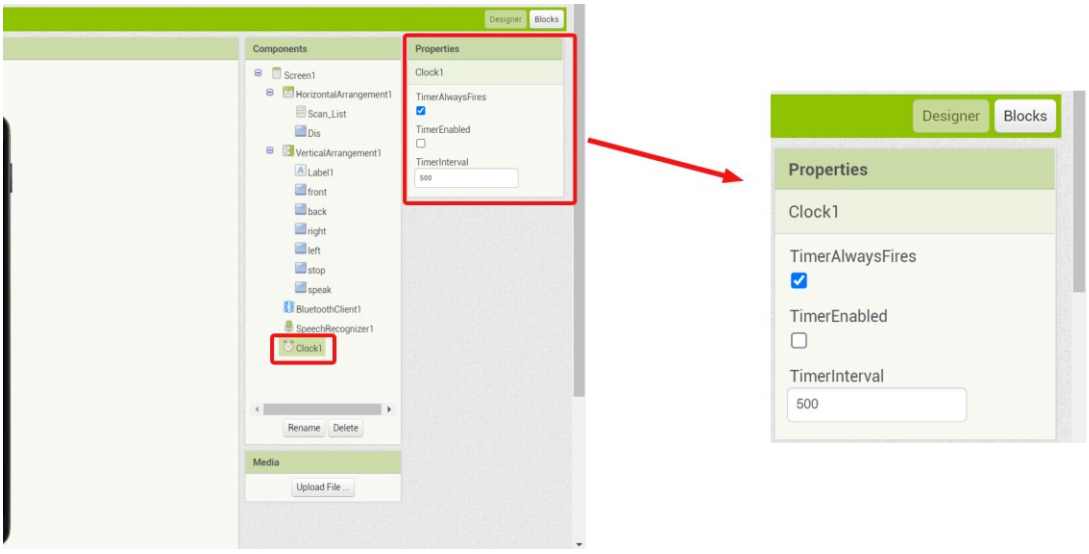
The user will need to press the button "Press to speak" only once, and only in the beginning of the process, in order to initialize the voice/speech recognition. When s/he do that, the application will start searching for valid voice commands. After a specific time interval (e.g., 2 seconds) the application will automatically search for a new voice command.

Let's see how we can do that:

1) Adding a Clock Sensor

Go again to the Designer menu and drag a "Clock" sensor from Sensors tab, to the application's designing area.

On the Clock's Properties check the TimeAlwaysFires box and set the TimeInterval to 2000 (milliseconds).



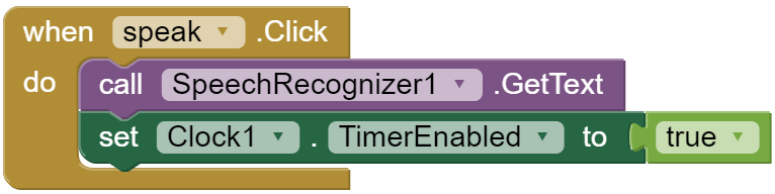
Co-funded by the European Union

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.

2) Modifying the application

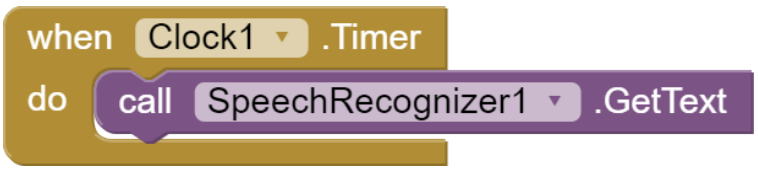
Now, go to the Blocks menu to modify some parts of the existing script while adding some more blocks of commands.

Inside the "when speak.Click" handler, add the set Clock1 TimerEnabled command and snap a true logic command on the right side of the block. This while enable our Clock to fire when the "Press to speak" button is pressed.



Now that we have fired/ignited the timer for the first time (after pressing the "Speak" button), the next step is to re-activate the SpeechRecognizer each time the countdown ends. The timer's duration is automatically renewed through this process.

To do that, we need to instruct our application to call the SpeechRecogniser to get the text of what it hears, each time the Clock's Timer fires. This can be realized by creating the following block of commands.



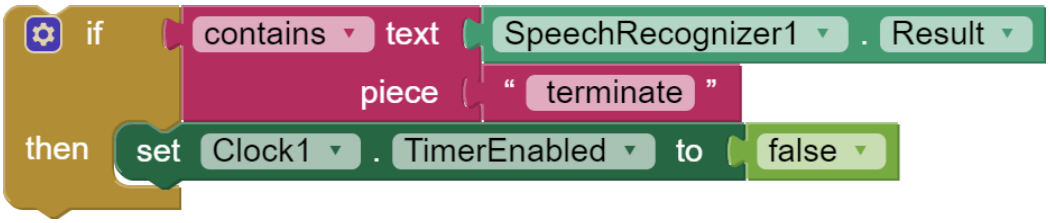
Let's test the application. *Again, go to the Build menu and generate the new version of the application.*

Does it properly work or is there any malfunction? Discuss with your team and write your answer below.

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.

Add the following block of commands to the "When SpeechRecognizer1.AfterGetting text", and test again the application.



Does the malfunction still exist?

Experiment further with the application to optimize the present solution. For example, try to change the duration of the time interval and write your observations on the table below. Keep in mind that the time is counted in milliseconds (1 sec = 1000 ms).

TimerInterval	Observation

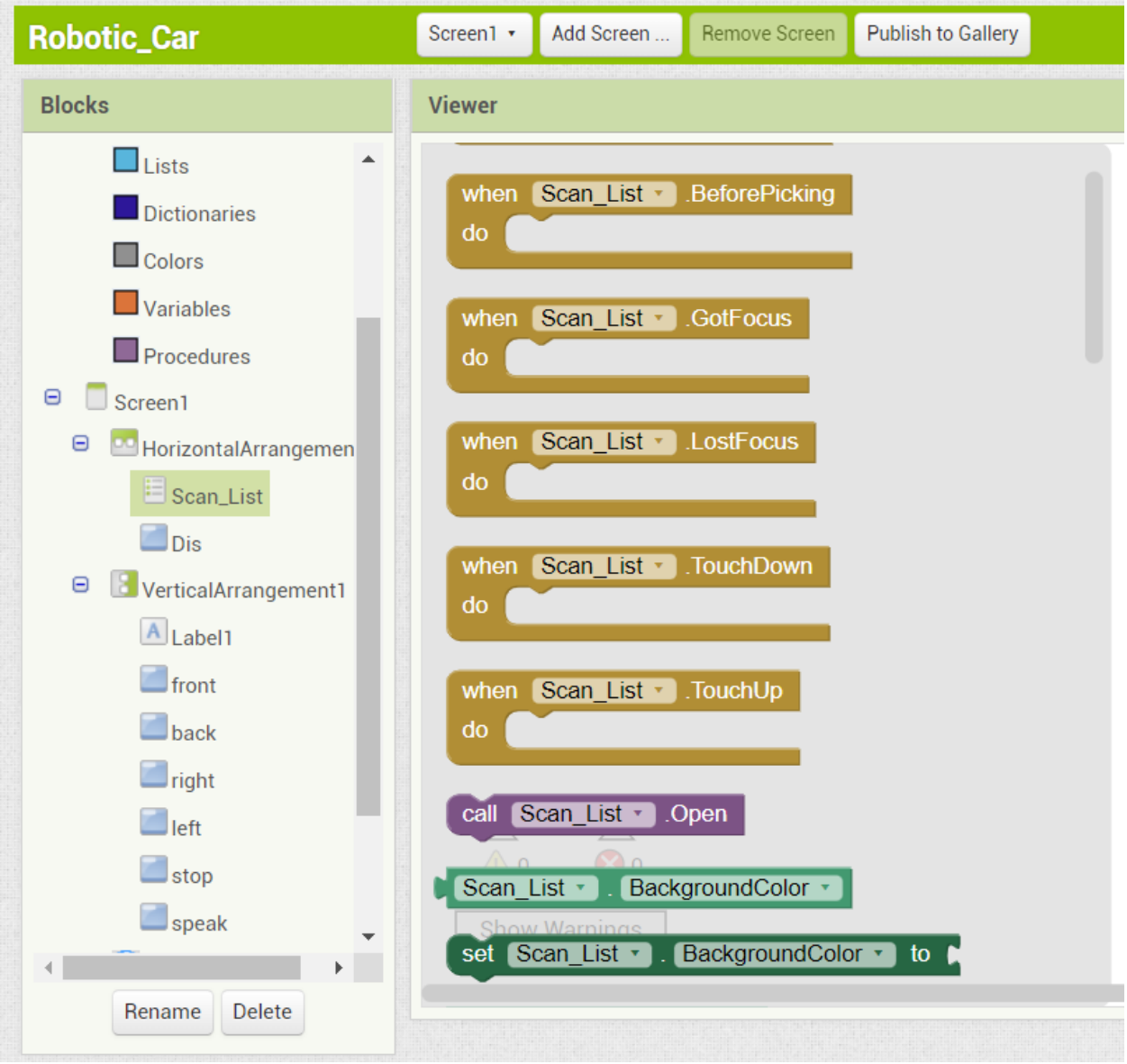
Co-funded by the European Union

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.

Tip Zone

Finding the commands:

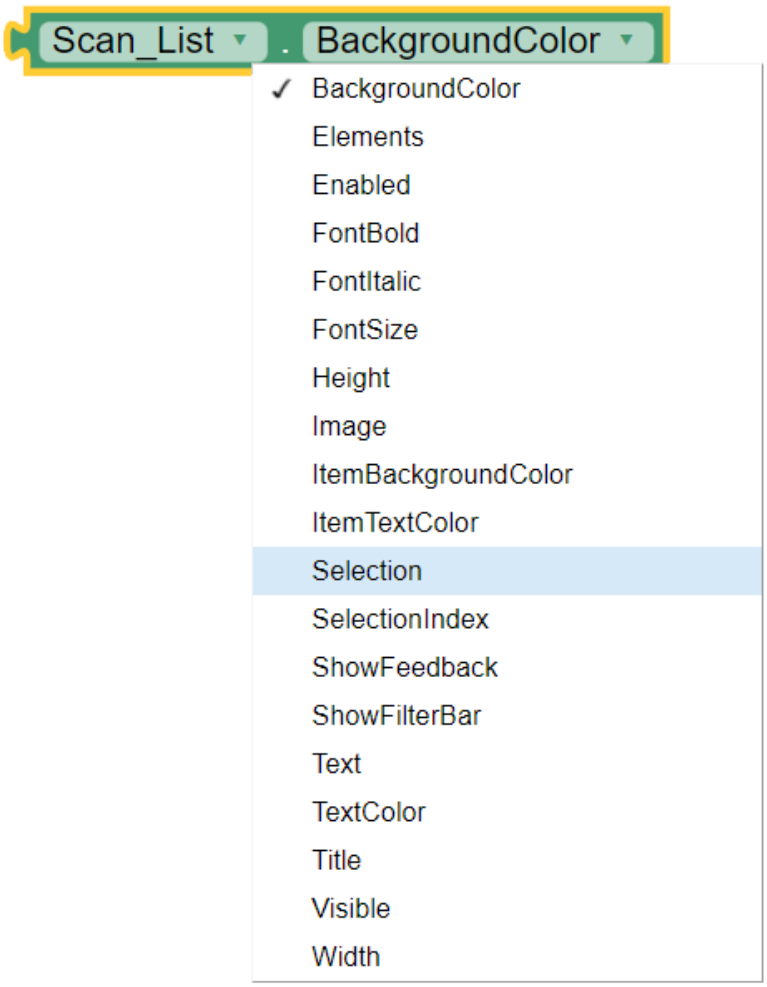
To find some of the needed commands, click on the corresponding item (i.e., Scan_List in the following example) and search on the popping up floating menu.



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.

Some commands contain more than one option. The following image presents such a command.



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.