# Arduino\_mega

#include imports a package or a file

Text

Description automatically generated

SoftwareSerial allows us to communicate through bluetooth devices or direct cable over a serial port. In this case, RX and TX (input/output).

arduino-timer is a package that allows us to run things in an interval or a time out. *So instead of using a loop, we can run another loop (interval) separate from a main loop that goes as quickly as it can. So it controls an interval based on the timer.*

Everything that ends with the extension .h, is a file or a package created by myself (loaded form the same folder).

“arduino-timer” is a library that allows use to run certain function in an interval, so for example we can run a function every 5 seconds. It also allows us to run a function in a delay, for example: run this function 10 seconds from now. It does this all without blocking the main loop function (“loop()”) which is run on the Arduino as fast as it can to run all the code.

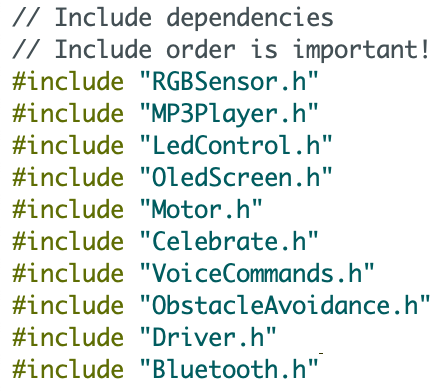
Text

Description automatically generated

Auto timer is set to timer\_create\_default(), this will create the timer instance on the timer variable.

We also create some global variables that we can use everywhere in our code:

* hasWon will determine if we just won, it will be set to true after winning so it will on trigger the “celebration” for winning once.
* voiceControlled will tell us if voiceControl is currently active, if this is true we will act different in some cases when the robot is driving on its own.



Here we are including all the other files. Every module is split its own file to make the code more clear to understand.

C++ (which is the coding language we use for the Arduino) reads code from top to bottom, so the order we import things is important!

As an example, the “OledScreen.h” file needs to use functions declared in “MP3Player.h”.

Text

Description automatically generated

Serial.begin(9600) where 9600 is the BAUDRate (this is the communication port use by the serial communication, in this case it’s the serial communication between the Serial monitor from the Arduino IDE and the Arduino itsself. We use this to output errors and information logs to the Serial monitor).

Setup() runs all the first functions you need for things to start up, so it initializes everything

All these functions are defined in the files assigned to the modules.

Graphical user interface, text, application

Description automatically generated

The “loop()” function is triggered by the Arduino, as the name suggests, in a loop as fast as it can. There is some code that can block the loop. Some examples are a “while” loop, “for” loop or a delay. It won’t be able to continue executing the loop untill these are done.

timer.tick() tells the timer it should start the count, this count is then used for intervals and timeouts. This way it can determine how much time passed and if it needs to trigger any of the timer tasks we defined.

“UpdateOledScreen()” triggers the function in “OledScreen.h” that will draw the correct bitmap image to the screen. This will be executed in the loop so it can play the talking animation as soon as the MP3Player plays an MP3 file.

If we haven’t won yet (!hasWon) and we detected the color red (colorRedDetected) and we are currently controlled by voice commands (voiceControlled) we will trigger the winner function. This will make the Robot dance and play the victory MP3 file defined in “Celebrate.h”

runBLECommunication(); is a function that will listen to any incomming data from the BLE module, this is used to communicate with the timer.

listenToVoiceCommand(); is a function that listens to any incomming voice commands that are send from the phone.

if(!handbrake) is a variable defined in “Motor.h”, when this is true we don’t want the arduino to drive forward. In the case handbrake is false, we do allow the robot to drive forward, we do this by triggering the “selfDriving()” function defined in “Driver.h”.