Technical analysis – Arduino code

Version 1.0



Contents

[1. Libraries 4](#_Toc482377974)

[1.1. SimpleTimer – www.playground.arduino.cc/Code/SimpleTimer 4](#_Toc482377975)

[1.2. Smartcar shield – www.github.com/platisd/smartcar\_shield 4](#_Toc482377976)

[1.3. NewPing – www.playground.arduino.cc/Code/NewPing 4](#_Toc482377977)

[1.4. LSM303 – www.github.com/pololu/lsm303-arduino 4](#_Toc482377978)

[1.5. Wire – built-in library 4](#_Toc482377979)

# Libraries

The sketch utilizes in total five different libraries during execution.

## SimpleTimer – www.playground.arduino.cc/Code/SimpleTimer

Due to Arduino’s single thread execution and limited performance, code that does not necessarily provide value by constantly being executed can instead be advantageously scheduled. Our sketch includes two instances of such code; sending battery voltage as well as sending sensor values (from ultrasonic sensors, distanced to objects). With SimpleTimer, these values are instead sent regularly to the phone application minimizing the work for the Arduino.

## Smartcar shield – www.github.com/platisd/smartcar\_shield

This project is based on the Smartcar platform provided by Dimitris Platis. It also utilizes his library, Smartcar shield for an easy and intuitive way of controlling the smartcar.

## NewPing – www.playground.arduino.cc/Code/NewPing

When developing the sketch, we found the NewPing library far superior to any other ways of getting values from the ultrasonic sensors in a fast and stable way. The NewPing library is used for all the ultrasonic sensors used in our SmartCar.

## LSM303 – www.github.com/pololu/lsm303-arduino

Even though we use a magnetometer from Adafruit, we use Polo’s library for getting the heading. The Polo library has shown to provide more accurate data and it also gives us the ability to calibrate the magnetometer for even more accurate readings.

## Wire – built-in library

Lastly, the Wire library is used by the magnetometer library, LSM303, to communicate with the magnetometer over the I2C**/TWI pins.**

# Methods

In accordance with the format of runnable Arduino code, our code is structured to include both a setup as well as a loop method. The setup method is automatically ran once at boot of micro controller before the loop method is infinitively executed. The loop method will run until the Arduino is powered off. Thus, all code that we want to become executed needs to somehow be called from the loop method.

## void setup()

As aforementioned, the setup method is required for creating a runnable Arduino code. Apart from initializing and specifying baud rates, this method also takes the calibration values for the magnetometer, line 78 and 79, as well as setting the interval for sending the battery voltage (every 5 second) and the sensor values (every 0.5 second) on line 82 and 83.

## void loop()