Motor Controller via NodeMCU and Desktop Application Using Java

By:

Osama Mohmed Hamdy
Karim Muhammed Said
Sarah Yasser Abdelaziz
Ahmed Hassan Hosny
Joseph Girgis Hosni

Supervised by: Eng Eman Hesham

6 Jan, 2022

Table of Content

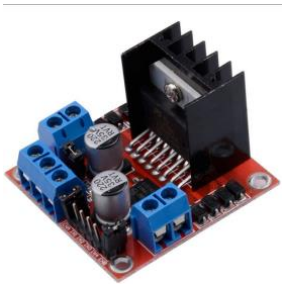
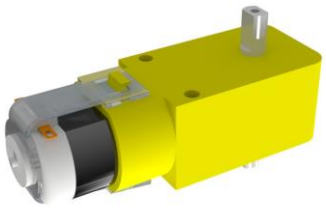
Abstract

The aim of this project is to control the speed of a DC-Motor using a desktop application, a desktop application was designed using Java and interfaced the application with a microcontroller (**NodeMCU**) which is used to control the DC-Motor and also control a Car if wanted.

Hardware

The Desktop Application is used to control a set of component which includes a **NodeMCU** and being connected to that desktop application through a JAR file (JserialCom).

1. The used Component and supplies of the Hardware:

Name	Figures
L298 Dual Motor Driver	
DC - Motor	
NodeMCU (ESP8266)	

2. The Hardware Connection:

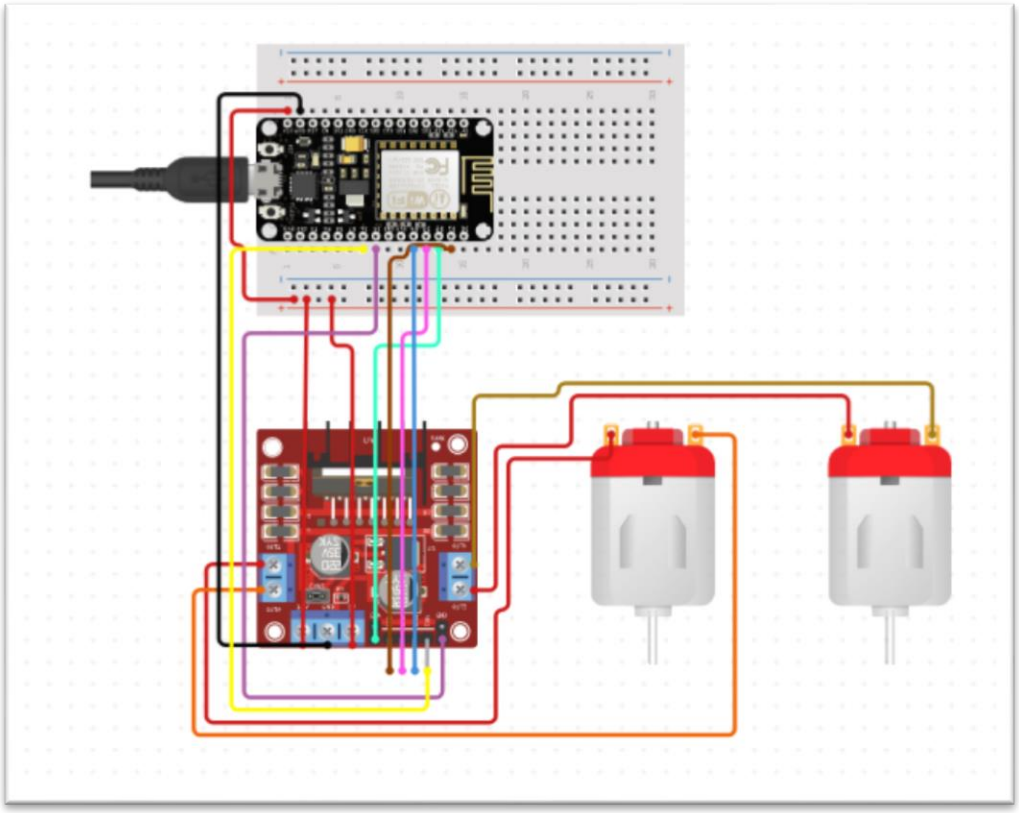


Figure 1.1: The Hardware Connection

Software

The software is divided into two parts: the **Java code** which implements the **Desktop Application** and the **NodeMCU Code** which is responsible of Controlling the **NodeMCU**.

1. Java Code

The Code implements several Scenes, each one of them is responsible of a different function; the code is made by JavaFX Application.

The GUI is starting with an intro which describes the Application.

Figure 2.1: The Intro

Then you will be transported to the main scene if you did click anywhere, in the main scene you will get to choose if you want to control the speed of a single motor (Motor Mode) or if you want to control a car you can choose (Car Mode)

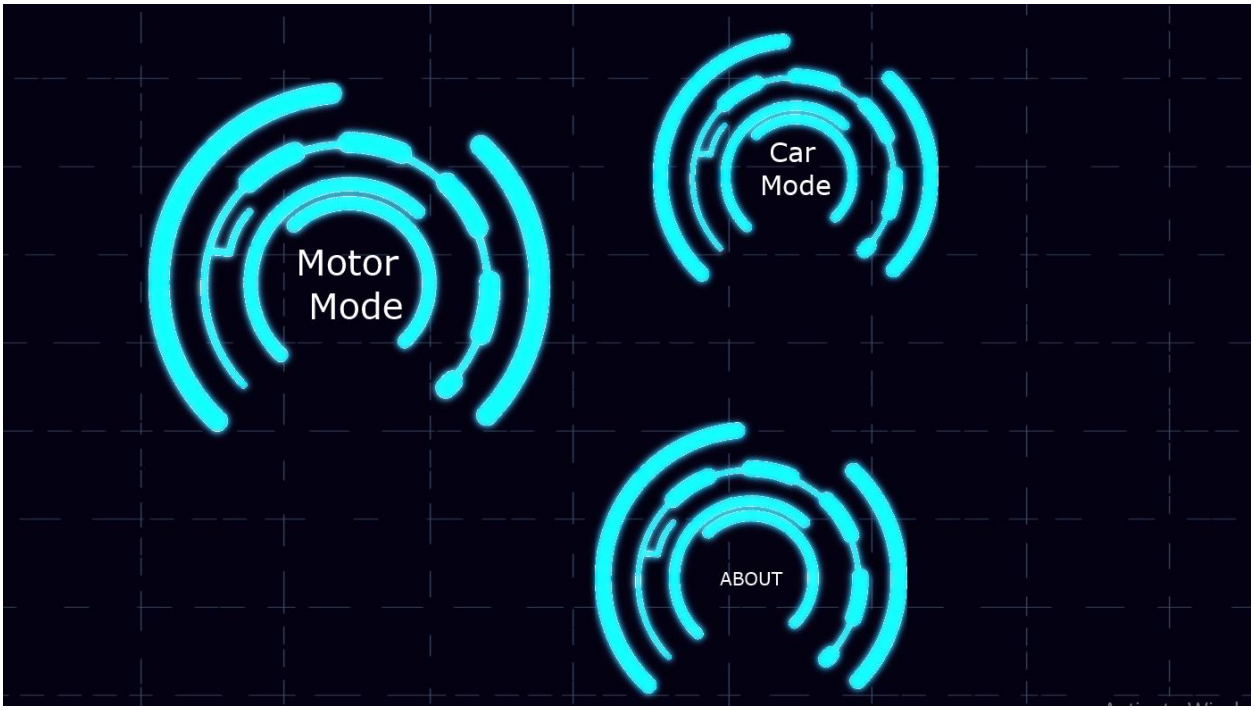


Figure 2.2: The Main Scene

1.1 Medusa Library

Medusa is a library that has useful classes ready to create and use Different types of Gauges in JavaFX application, all you should do is install its JAR file, add it to the application's libraries and import it to your GUI.

```
/************************/
/*        Gauge         */
/************************/

gauge = new Gauge();
gauge.setSkin(new ModernSkin(gauge));
gauge.setUnit("Rpm");  //unit
gauge.setUnitColor(Color.WHITE);
```

Figure 2.3: Sample of Gauge Code by Medusa Library

Medusa gives you the ability to control the Gauge's Skin, for example the following figure (Fig (2.4)) is the Skin of the gauge we used.

Figure 2.4: ModernSkin of Gauge from Medusa Library

1.2 JserialCom Library

The JserialCom library is used for making a connection between java code and the NodeMCU code also sending and receiving data.

```
/** public constructor used to start the serial communication **/
public Connect(){
    comPort = SerialPort.getCommPort("COM5");
    comPort.openPort();
    comPort.setComPortTimeouts(SerialPort.TIMEOUT_READ_SEMI_BLOCKING, 0, 0);
}
```

Figure 2.4: Sample of Code from JserialCom Library