|  |  |  |
| --- | --- | --- |
| Logo__SSPU_2016_Barva | | |
| **Závěrečná studijní práce**  **dokumentace** | | |
| **Auto na dálkové ovládání** | | |
| Lukáš Baďura | | |
| IMG_3240 | | |
|  | |  |
| **Obor:** | 18-20-M/01 INFORMAČNÍ TECHNOLOGIE  se zaměřením na počítačové sítě a programování | |
| **Třída:**  **Školní rok:** | IT4  2019/2020 | |

#### Poděkování

*Děkuji panu učiteli Ing. Petru Grussmannovi za užitečné rady, vylepšení a konzultaci mého projektu. Mým spolužákům za pomoc a řešení společných problémů.*

Prohlašuji, že jsem závěrečnou práci vypracoval samostatně a uvedl veškeré použité   
informační zdroje.

Souhlasím, aby tato studijní práce byla použita k výukovým účelům na Střední průmyslové   
a umělecké škole v Opavě, Praskova 399/8.

V Opavě 31. 12. 2019

*podpis autora práce*

**ANOTACE**

Cílem projektu bylo sestavení auta na dálkové ovládání z mobilní aplikace v Android Studiu pomocí WiFi. Hardwarovou část tvoří vývojová deska NodeMcu s čipem ESP8266. Zařízení za pomoci již zmíněného ESP8266 a modulu L298N řídí čtyři převodové motory. Mobilní aplikace posílá ESP hodnoty, pokud jsou vyhodnoceny jako „pravda“ je spuštěna příslušná funkce pro pohyb auta. Softwarová část auta je řešena v jazyce Arduino, který je kombinací jazyků C a C++.

Goal of the project was build a remote controlling car with mobile application in Android Studio which uses WiFi to communicate. Hardware part of the project is development board with chip ESP8266. Project uses ESP8266 and L298N module to control 4 geared motors. Mobile application sends values to ESP. If is value evaluated as „true“ the function for car direction controll is launched. Software part of the project is wrote in Arduino language which is the combination of C and C++ languages.

**Klíčová slova**

WiFi; dálkové ovládání; NodeMcu; Arduino; převodové motory; mobilní aplikace; motorový modul

OBSAH

[Úvod 5](#_Toc29845323)

[1 SESTAVENÍ AUTA 6](#_Toc29845324)

[2 Využité technologie 7](#_Toc29845325)

[2.1 Hardware 7](#_Toc29845326)

[2.1.1 Seznam součástek 7](#_Toc29845327)

[2.1.2 NodeMcu 8](#_Toc29845336)

[2.1.3 H-můstek L298N 8](#_Toc29845337)

[2.2 Software 9](#_Toc29845338)

[2.2.1 Visual Studio Code 9](#_Toc29845339)

[2.2.2 App Inventor 9](#_Toc29845340)

[3 ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY 10](#_Toc29845341)

[3.1 Schéma 10](#_Toc29845342)

[3.2 Použité knihovny 11](#_Toc29845343)

[3.2.1 ESP8266WiFi.h 11](#_Toc29845344)

[3.2.2 WifiClient.h 12](#_Toc29845345)

[3.2.3 ESP8266WebServer.h 12](#_Toc29845346)

[4 HARDWAROVÉ ZAŘÍZENÍ 13](#_Toc29845347)

[4.1.1 Popis fungování ovládání auta 13](#_Toc29845348)

[4.1.2 Připojení k WiFi 15](#_Toc29845349)

[4.1.3 Převod stringových hodnot z aplikace 16](#_Toc29845350)

[4.2 Mobilní aplikace 16](#_Toc29845351)

[5 VÝSLEDKY ŘEŠENÍ 18](#_Toc29845352)

[5.1 Podoba zařízení 18](#_Toc29845353)

[5.2 Podoba mobilní aplikace 19](#_Toc29845354)

[5.3 Problémy 20](#_Toc29845355)

[Závěr 21](#_Toc29845356)

[Seznam použitýCH INFORMAČNÍCH ZDROJů 22](#_Toc29845357)

Úvod

Tento projekt jsem si vybral, aby mi závěrečná práce nebyla nejen znalostním přínosem, ale i zábavou.

Auto na dálkové ovládání mě zaujalo a vybral jsem si ho, po shlédnutí mnoha videí spojených s roboty a auty ovládanými přes WiFi.

Projekt má dvě hlavní části. Ručně sestavené auto na dálkové ovládání a mobilní aplikaci. Tyto dvě části vzájemně komunikují pomocí již zmíněné WiFi. Ve své dokumentaci popisuji použité technologie a principy fungování.

# SESTAVENÍ AUTA

Po zakoupení součástek jsem začal se zapojením jednoho motorku s kolem do jednoduchého obvodu s ESP8266, H-můstkem L298N a externím napájením.

Nahrál jsem do obvodu jednoduchý program pro roztočení kola s motorkem vpřed. Po úspěšném testu součástek jsem se rozhodl na každý motorek s kolem připájet dva drátky a následně přilepit tavnou pistolí ke spodní části podvozku.

Dále jsem na vrchní část podvozku nalepil postupně ESP8266, H-můstek L298N a externí zdroj napájení. Motorky jsem pomocí připájených drátků zapojil do výstupních konektorů H-můstku. H-můstek jsem propojil kabely s ESP a následně zapojil externí napájení. Po zapojení obvodů již na podvozku jsem přešel k nahrání jednoduchého kódu pro pohyb auta vpřed a vzad.

Po úspěšném otestování jsem přešel k finálnímu doladění celého obvodu. Drátky z H-můstku jsem připájel na piny ESP pro bezpečnější funkčnost při jízdě, k externímu napájení jsem přidal vypínač pro pohodlnější zapnutí a drátky vedoucí z motorků jsem lépe spojil k sobě, pro lepší stabilitu a funkčnost auta. Po uspokojivém zapojení všech součástek jsem přešel k řešení problematiky softwarové části.

# Využité technologie

## Hardware

### Seznam součástek

### Vývojová deska NodeMcu

### H-můstek pro krokový motor L298N

### Dvě 18650 Li-Ion Baterie

### Bateriový box

### Propojovací kabely F-F

### Propojovací kabely M-M

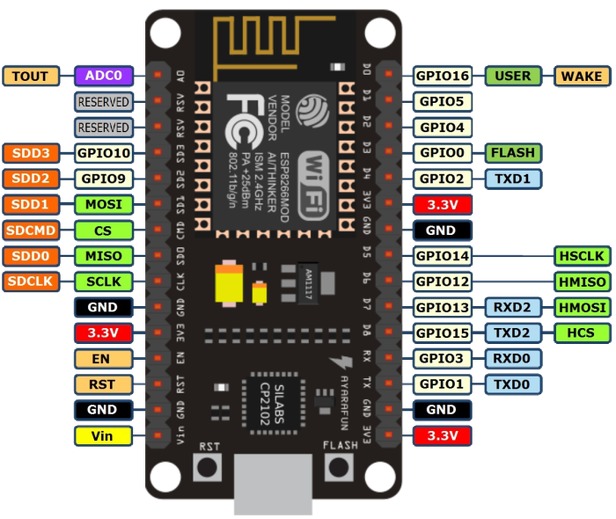
### Čtyři kola s převodovými motory

### Přepínač 6A 250VAC

### NodeMcu

Základ projektu tvoří vývojová deska NodeMcu. Deska je založena na čipu ESP8266 od společnosti Expressif. Hardware je založen na modulu ESP-12E.

Deska obsahuje 11 digitálních I/O pinů a 1 analogový vstup. Programu je k dispozici paměť o velikosti 128kB. NodeMcu na rozdíl od ostatních součástek pracuje s napětím 3,3V.



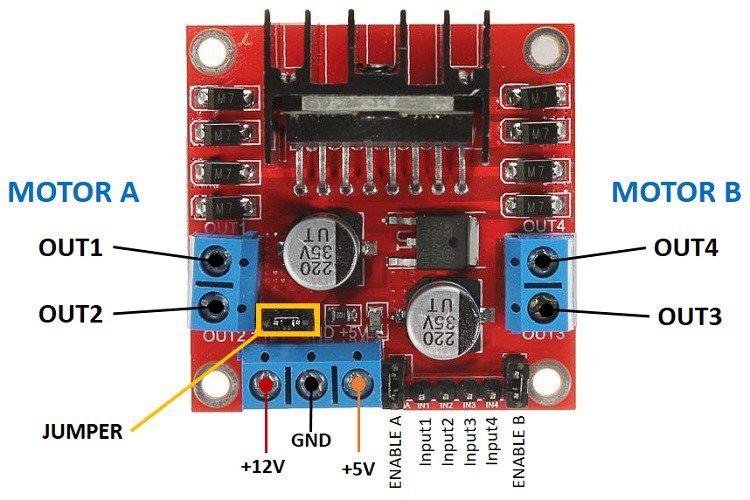
### 

### H-můstek L298N

Tento modul používá ST L298N čip. Čip disponuje dobrou schopností řízení, nízkou teplotou a silnými anti-interferenčními schopnostmi.

Slouží k řízení krokových motorů, nebo jako v mém případě převodových motorů. Dá se napájet až do 46V.

Výstupní konektory pro zapojení motorů jsou po stranách označeny jako OUT1, OUT2, OUT3 a OUT4. Dále modul disponuje piny ENABLE A a ENABLE B, které slouží pro aktivaci motorů zapojených do již zmíněných výstupních konektorů. Piny IN1, IN2, IN3 a IN4 slouží pro řízení motorů. Na modulu se nachází 12V silové napájení pro motory, 5V napájení logiky modulu a GND uzemnění.



## Software

### Visual Studio Code

Jako vývojové prostředí pro můj projekt, jsem zvolil Visual Studio Code. Prostředí jsem si vybral z důvodu, protože jsem s ním již měl drobné zkušenosti, je vizuálně hezké, přehledné a hlavně je zde možnost použití rozšíření Platformio, které je nezbytné pro můj projekt a velice usnadňuje práci.

### App Inventor

Jako vývojové prostředí pro moji mobilní aplikaci jsem si zvolil grafický programovací jazyk App Inventor.

V jazyce se dají vyvíjet plnohodnotné aplikace, a jelikož jazyk podporuje operační systém Android, na kterém funguje můj mobilní telefon byl ideální volbou.

# ZPŮSOBY ŘEŠENÍ A POUŽITÉ POSTUPY

## Schéma

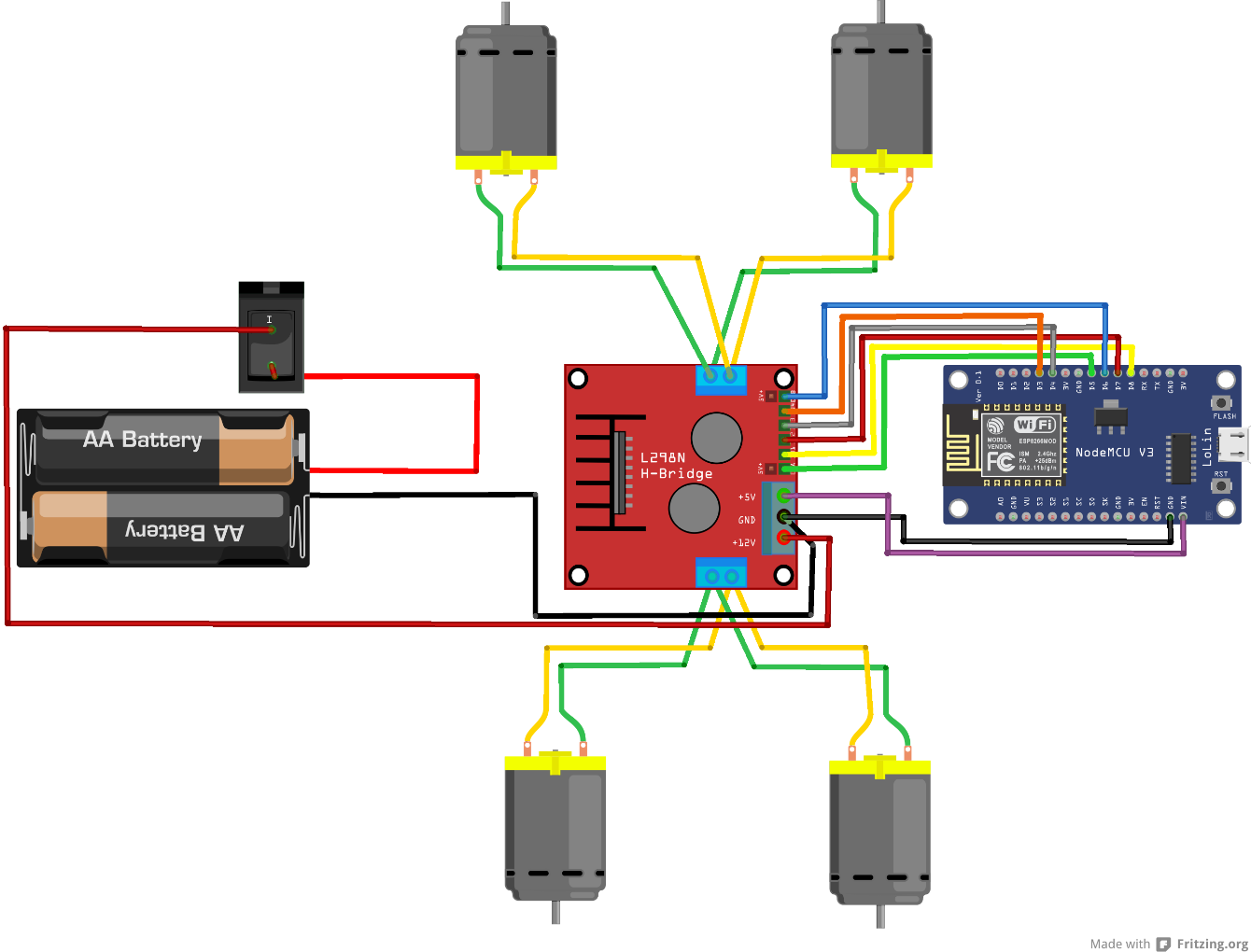


Schéma jsem vytvořil v programu Fritzing. Program je open-source systém, který umožňuje v jednoduchém rozhraní graficky sestavit zapojení všech komponentů. Fritzing jsem zvolil, protože je velice přehledný, jednoduchý a splňuje veškeré požadavky, které jsem potřeboval. Výsledné schéma je díky grafickému zpracování velmi jednoduché k pochopení.

## Použité knihovny

### ESP8266WiFi.h

Knihovna pro připojení ESP8266 modulu k WiFi. Pro lepší představu níže přikládám výstřižek z mého kódu.

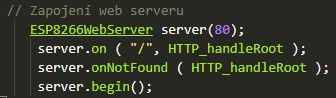


### WifiClient.h

Tato knihovna umožňuje vývojové desce připojení k internetu. Může sloužit jako server přijímající příchozí připojení nebo jako klient pro odchozí. Knihovna podporuje šifrování WEP a WPA2 Personal, nikoli však WPA2 Enterprise.

### ESP8266WebServer.h

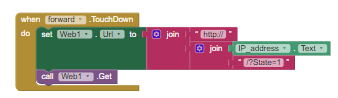
Knihovna nutná pro práci s web serverem na našem čipu ESP8266. Knihovna ví jak zpracovat požadavky http, jako je GET a POST. Také dokáže podporovat pouze jednoho klienta. Níže přikládám krátký výňatek z mého kódu, kde spouštím web server.



# HARDWAROVÉ ZAŘÍZENÍ

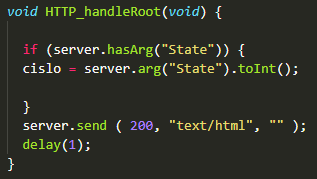
### Popis fungování ovládání auta

Auto je ovládáno mobilní aplikací napsanou v jazyce Scratch.

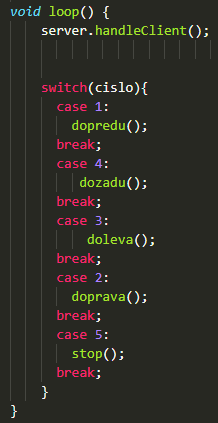


Zde je příklad tlačítka sloužící pro pohyb auta vpřed. Celý blok funguje následovně, pokud je ztisknutý symbol šipky vpřed v mobilní aplikaci, tak se do ESP pošle hodnota “/?State=1“.

Následně hodnota pokračuje do funkce v ESP, která zjistí, jestli přišel parametr “State”. Pokud parametr opravdu přišel, uloží se parametr “State” do proměnné číslo, která je intové hodnoty, protože parametr “State”, který je mimo jiné strinogové hodnoty, je funkcí toInt(); převeden do hodnoty intové, aby mohl být snadněji a přehledněji porovnán.

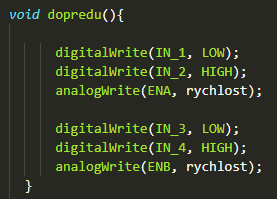


Když je toto všechno splněno, hodnota uložená v proměnné číslo, je poslána do cyklu switch.



Každá šipka v aplikaci posílá jinou hodnotu od 1 do 5. Podle toho switch rozpozná, jakou funkci pohybu má následně vyvolat.

Hodnota 1 pro zmíněnou funkci forward je vyhodnocena switchem, jako pravda a je vyvolána funkce “dopredu”.



Funkce “Dopredu ” sepne na určitých pinech a rozpohybuje kola daným směrem. Proměnná rychlost je zde nastavena pevně na hodnotu 800. Tenhle proces se opakuje stále dokola, při stisknutí tlačítek v mobilní aplikaci.

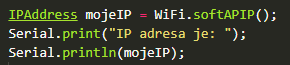
### Připojení k WiFi

V následující kapitole jsem popsal principy fungování a postupy připojení mého ESP8266 k WiFi síti.

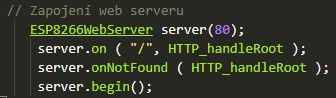
ESP8266 je schopno díky knihovnám vysílat vlastní WiFi, kterou mám uloženou v konstatní proměnné ssid s názvem “Auticko”.



Po nahrání programu do auta, se na sériové lince vypíše IP Adresa ESP, díky následujícím příkazům, které do proměnné mojeIP uloží skutečnou IP Adresu ESP a následně je příkazem vypsána na sériové lince.



Zapojení WebServeru je řešeno následnými třemi jednoduchými příkazy.

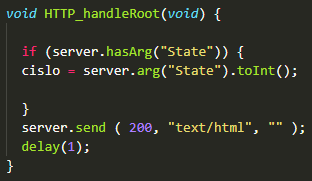


### Převod stringových hodnot z aplikace

V mojí mobilní aplikaci je každé jednotlivé tlačítko pro pohyb auta uloženo a posíla do ESP jinou hodnotu. Hodnoty jsou uloženy ve stringových proměnných “State”.

Když ESP otestuje podmínkou že opravdu přijalo hodnotu “State”, je hodnota následně převedena funkcí toInt(); na hodnotu číselnou a uložena do proměnné číslo.

Proměnná číslo se dále porovná v cyklu switch a vyvolá se daná funkce pro pohyb auta ve správném směru.



## Mobilní aplikace

Mobilní aplikaci jsem psal ve vývojovém prostředí App Inventro v jazyce Scratch. Tento jazyk s prostředím jsem zvolil, díky nedostačujícím znalostem Javy, jednoduchosti, přehlednosti jazyku a celkovému doporučení všech podobných githubových projektů.

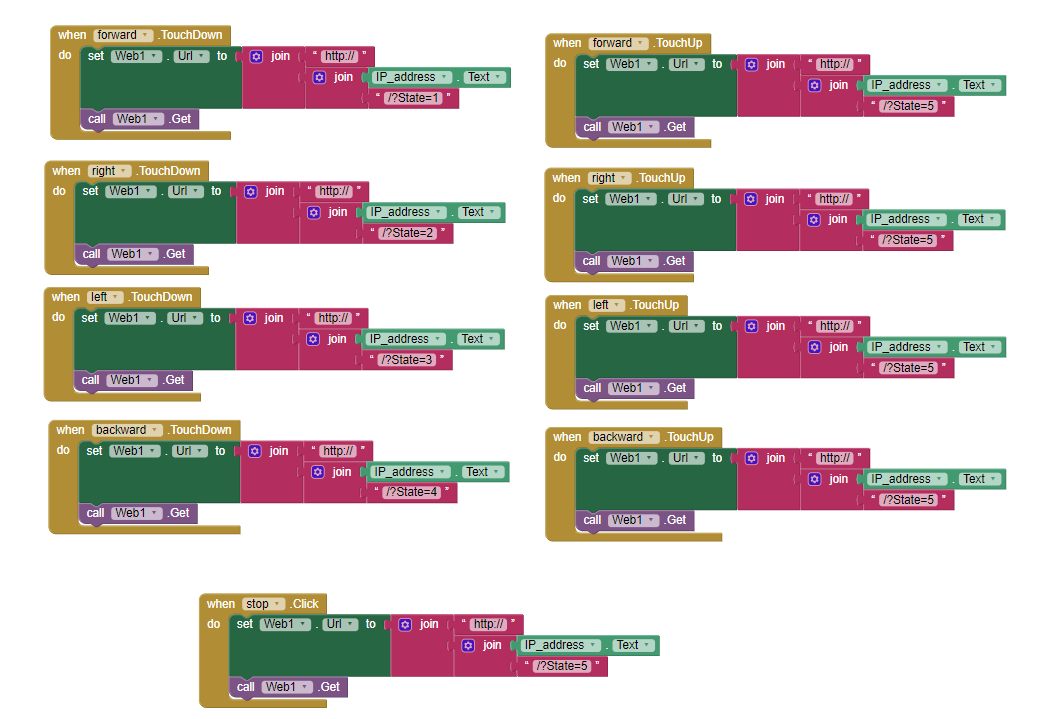
Aplikace je poměrně jednoduchá, jak po zdrojové stránce, tak i po stránce grafické.

Zdrojová struktura aplikace se skládá z několika bloků pro pohyb danými směry, konkrétně se jedná o tlačítka dopředu, dozadu, vlevo, vpravo a tlačítko stop. Všechny tlačítka fungují na stejném principu.

Pokud je stisknutá šipka na obrazovce telefonu, aplikace pošle do ESP hodnotu, se kterou se dále pracuje v programu pro ESP.

Každé tlačítko je doplněno také o hodnotu, která se odešle při spuštění tlačítka, aby auto zastavilo a na jedno stisknutí nepokračovalo dále ve své trase.

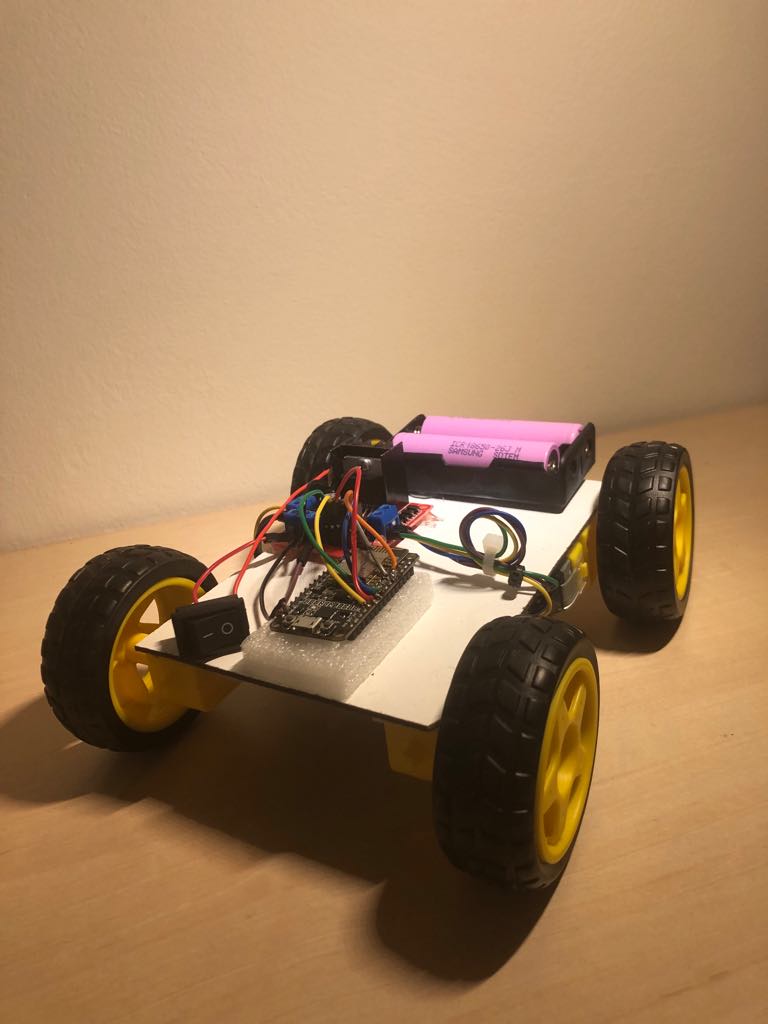
Grafická část je velmi jednoduše tvořena čtyřmi šipkami a tlačítkem stop, které byly tvořeny programem Gimp. Aplikace je doplněna o logo školy a tabulku, kde je od základu vepsaná IP Adresa ESP.



# VÝSLEDKY ŘEŠENÍ

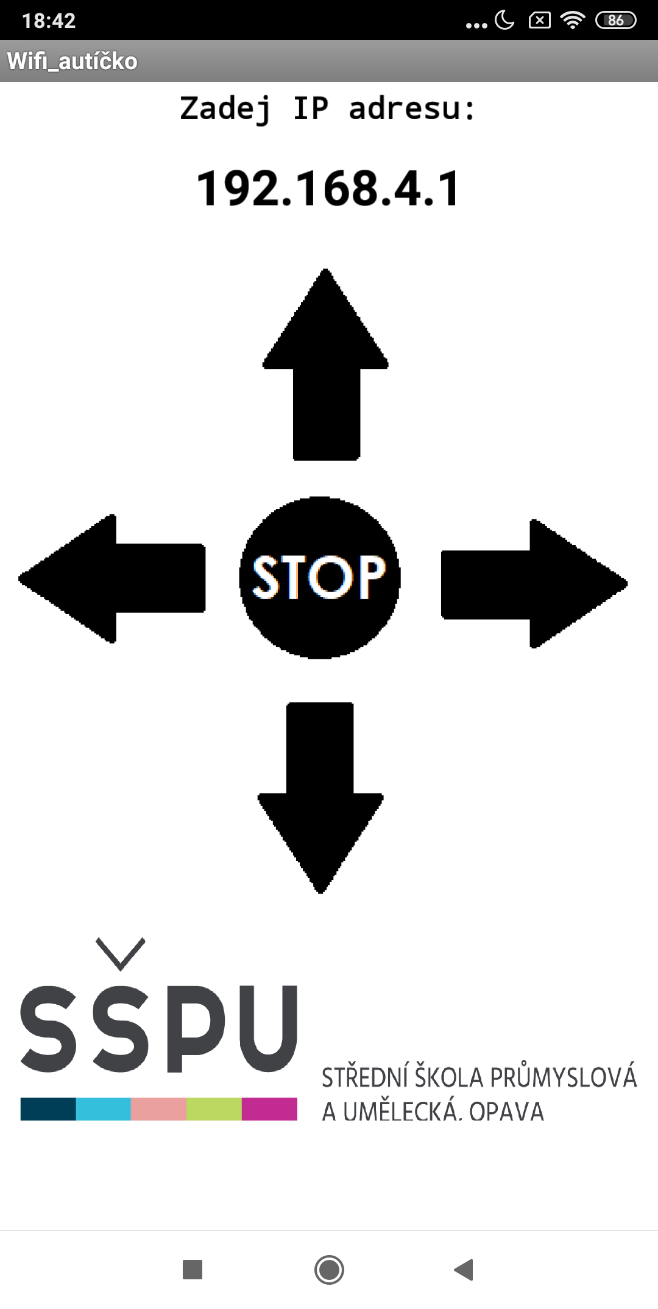
## Podoba zařízení

S autem jsem vcelku spokojen, do budoucna bych chtěl přidat senzor proti naražení do překážky a karosérii pro lepší vzhled. Prozatím auto mé očekávání splňuje a je plně funkční, což bylo hlavním cílem mého projektu.



## Podoba mobilní aplikace

Mobilní aplikace splňuje svůj účel. Je velmi jednoduchá a naprosto dostačující pro běžného uživatele. Do budoucna bych rád aplikaci vylepšil o akceleraci, lepší grafické zpracování a přidání bočních směrů, aby nebylo nutné mačkat dvě šipky najednou.



## Problémy

Při tvorbě svého projektu jsem se nepotýkal s mnoha problémy.

Pár jich však nastalo ze začátku díky chybnému připájení, se levé přední kolo točilo opačným směrem, než mělo. Tím samozřejmě brzdilo chod celého auta, jako celku.

Při programování ESP se mi mnoho problému nenaskytlo, protože můj kód je vcelku jednoduchý a funkční. Chvíli jsem bojoval s myšlenkou, jak dostat hodnoty do switche a celé převést. Vymýšlel jsem zbytečně složité věci, když řešení bylo velmi jednoduché.

Velký problém pro mě byla aplikace. Při představě, že se budu muset naučit Javu tak, abych byl schopen naprogramovat aplikaci pro řízení směru, která komunikuje s WiFI.

Po mnoha pokusech, přečtení mnoha githubových projektů, jsem dospěl k závěru, že snazší a efektivnější bude zvolit Scratch.

Při dokončování projektu jsem se potýkal již jen s banalitami, hlavně při přenášení do školy, které způsobilo neustále závady ne mém projektu, díky čemuž jsem byl nucen objednat nový motorek, napájet drátky několikrát znovu a mnoho dalšího.

# **Závěr**

Můj projekt splnil vše, co jsem od něj očekával. Bohužel mé očekávání nepředčil a nebyl jsem schopen splnit některé cíle. Do budoucna bych určitě rád zapracoval na svých znalostech, abych mohl podobné projekty postavit a vylepšit bez větších problémů.

Projekt mi byl přínosem, lépe jsem pochopil některé principy, fungování a celkově mi rozšířil obzory. Projekt po svém odprezentování věnuji mladšímu bratrovi jako hračku, protože to je zatím jeho jediné využití.

Seznam použitýCH INFORMAČNÍCH ZDROJů

[1] *LASKARDUINO.cz* [online]. eshop [cit. 2020-01-13]. Dostupné z: https://www.laskarduino.cz/. Závěrečný projekt. SŠPU Opava. Vedoucí práce Ing. Petr Grussmann.

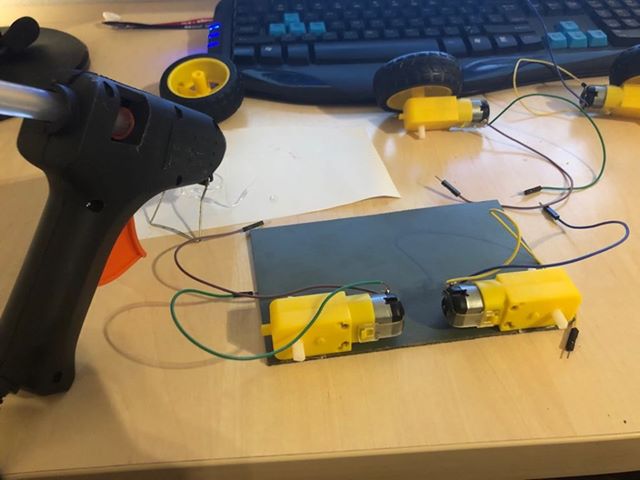
[2] *Github: social network* [online]. [cit. 2020-01-13]. Dostupné z: https://github.com/. Závěrečný projekt. SŠPU Opava. Vedoucí práce Ing. Petr Grussmann.

[3] *Arduino: community forum* [online]. [cit. 2020-01-13]. Dostupné z: https://forum.arduino.cc/. Závěrečný projekt. SŠPU Opava. Vedoucí práce Ing. Petr Grussmann.

[4] *Návody: Arduino* [online]. [cit. 2020-01-13]. Dostupné z: https://navody.arduino-shop.cz/navody-k-produktum/. Závěrečný projekt. SŠPU Opava. Vedoucí práce Ing. Petr Grussmann.

[5] *Youtube: Tutoriály* [online]. [cit. 2020-01-13]. Dostupné z: https://www.youtube.com/. Závěrečný projekt. SŠPU Opava. Vedoucí práce Ing. Petr Grussmann.

**Fotodokumentace**

****



