TF ČZU

Autíčko ovládané přes Bluetooth

Dokumentace k semestrálnímu projektu

Jakub Ryčl, Jan Lešetický

8. 1. 2015

Obsah

1. Úvod

V semestrálním projektu je řešen návrh softwaru a hardwaru dálkově ovládaného autíčka přes bluetooth z aplikace mobilního telefonu vybaveného OS Android. Dokumentace je rozdělena do tří částí. První z nich pojednává o fyzických parametrech a úpravách na samotném modelu autíčka. V druhé části (zahrnuje kapitoly 3, 4 a 5) je popsána a vyobrazena použitá elektronika spolu se schématy zapojení jednotlivých součástek a arduino dílů. Dále je zde umístěn popis zdrojového kódu a přehled financí za jednotlivé komponenty. Poslední část (kapitola 6) je určena mobilní aplikaci pro OS Android. Pro porozumění jsou nejdříve uvedeny základy programování v prostředí App Inventor 2, ve kterém je celá aplikace napsána. Následně je vysvětlen celý zdrojový kód aplikace.

2. Fyzické parametry autíčka

Pro semestrální projekt bylo zvoleno autíčko, které mělo odpovídající velikost a hmotnost, bylo vhodné, aby se dalo využít co největší množství komponent bez nutnosti opravy nebo dokoupení.

I přes to, že bylo autíčko zvoleno náhodně na inzerát „Prodám starší autíčko, dříve na dálkové ovládání“ (Obrázek 1) a i jeho cena byla již předem alarmující (30 Kč), ukázalo se nakonec jako velmi vhodné – jak po velikostní tak výkonové stránce.

Model autíčka má délku 260mm, šířku 110 mm a výšku 80 mm.

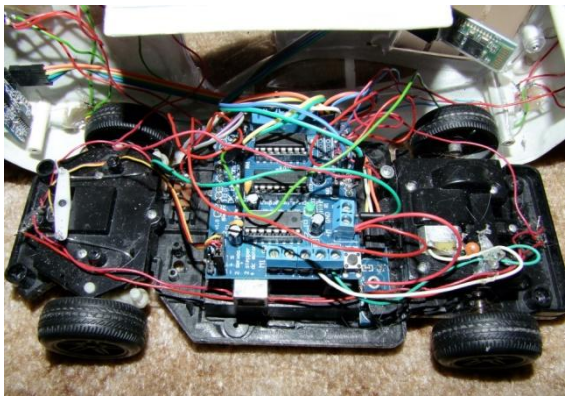
Obrázek 1 Inzerát na koupi autíčka

Autíčko bylo pro naše potřeby nutné upravit po všech stránkách, jednalo se zejména o:

- vyjmutí řídící jednotky,
- vyjmutí sedadel,
- vyjmutí antény,
- vyjmutí osvětlení,
- vyjmutí DC motoru pro ovládání natočení předních kol + vyjmutí aretační pružiny.

Po těchto krocích bylo autíčko připraveno pro nainstalování našeho vlastního hardwaru, zbylo na něm pro úplnost:

- podvozek,
- kastle autíčka,
- mechanika ovládání předních kol,
- mechanika zadních kol + DC motor
- kastle pro vložení 5 AA baterií
- on/off přepínač.



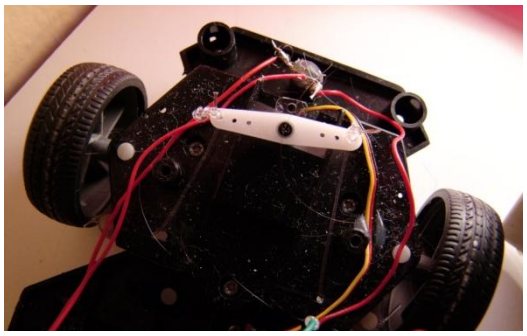
Obrázek 2 HW autíčka

Po navržení a otestování jednotlivých komponent byla vložena všechna elektronika (Obrázek 2) do modelu autíčka. Ve středové části je umístěno Arduino UNO. K upevnění k podvozku modelu byla použita lepicí tyčinka, která byla aplikována pomocí tavné pistole. Dále je v zadní části umístěn piezo bzučák, který plní funkci klaksonu. Jednotlivé komponenty a jejich schéma zapojení jsou popsány v kapitole 3 Součástky a komponenty.

2.1.Podvozek, ovládání autíčka a ostatní prvky

Vzhledem k tomu, že bylo vhodné, aby mělo autíčko možnost zatáčet ve větší rozlišovací schopnosti než pouze vpravo/vlevo, což umožňovalo původní řešení, bylo rozhodnuto, že se na ovládání natáčení přední nápravy použije servomotor.

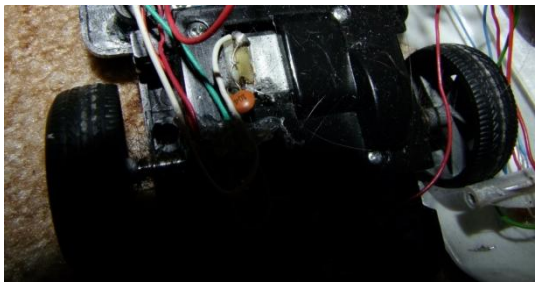
2.1.1. Zavěšení přední nápravy (fyzické uspořádání)



Obrázek 3 Přední náprava

Servomotor byl nejdříve otestován, v jakém bodě se nachází nulová pozice a pozice 180 neboli maximální. Následně byla na servomotoru nastavena pozice 90, která je středem rozsahu otáčení osy servomotoru a byla nasazena bílá páka (obr. 3). Dále bylo zapotřebí vyvrtat otvory pro vlasec, který je z jedné strany uvázaný za páku servomotoru a druhý na konci plastového dílu který spojuje obě kola.

2.1.2. Zadní náprava



Obrázek 4 Zadní náprava

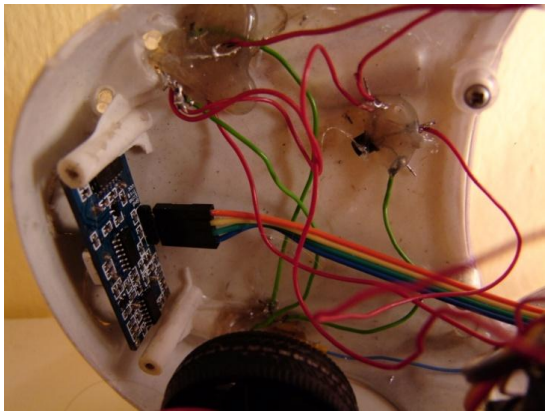
Zadní náprava (obr. 4) zůstala v původním stavu a je poháněna DC motorkem. Ten je zapojen přímo na motor shield, který zajišťuje změnu polarity na konektorech.

2.1.3. Napájení

Obrázek 5 Držák baterii

Celé autíčko je napájeno pěti tužkovými AA bateriemi, které dávají součet 7,5V (Obr. 5). Napájení je přivedeno na svorkovnici, která je umístěna na motor shieldu a tím je napájena všechna elektronika autíčka. Doba výdrže je dána kvalitou použitých baterii a na povrchu, na kterém se model pohybuje. Průměrná doba se pohybuje okolo patnácti minut, při plném používání osvětlení a povrhu s velkým koeficientem valivého tření na kola autíčka (např. koberec).

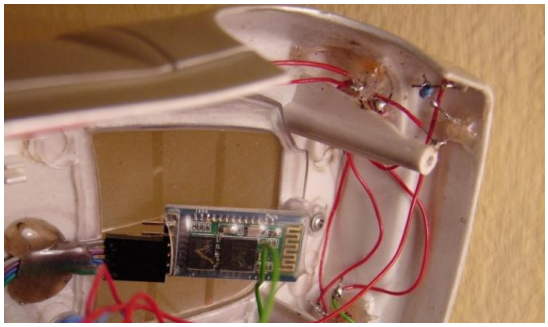
2.1.4. Ultrazvuk a bluetooth



Obrázek 6 Umístění utrazvuku

Pro umístění ultrazvuku bylo nutné vytvořit v přední části automobilu dva otvory o průměru 15 mm. Dále bylo nutné částečně ubrousit plastovou výztuž distačních sloupků, které mají závit a slouží k upevnění podvozku ke kastli modelu. K upevnění byla použita lepicí tyčinka, která byla aplikována pomocí tavné pistole. Dále je na Obrázek 6 ukázáno zapojení a upevnění LED diod, které slouží jako světlomety a směrová světla.

Na Obrázek 7 se nachází umístění bluetooth modulu. Záměrně byl umístěn v zadní části vozu, z důvodu nasměrování signálu směrem k mobilnímu přístroji. Dále z důvodu informativního a estetického. Bluetooth je osazen indikační diodou, která při zapnutí modelu rychle bliká a informuje o tom, že je autíčko připraveno ke spárování s mobilní aplikací. Po připojení se rychlost blikání diody výrazně sníží a informuje o tom, že je uživatel připojen. Po odpojení dioda opět signalizuje, že modul je opět připraven ke spárování.



Obrázek 7 Umístění modulu bluetooth

2.1.5. Vzhled modelu po úpravách

Na Obrázek 9 je vidět usazení ultrazvukového sensoru, který je použit jako bezpečnostní prvek proti přímému nárazu. Dále jsou zde ukázány zapnuté světlomety a podsvícení podvozku modelu automobilu. Na Obrázek 8 je pohled na zadní část modelu automobilu. V této části byly vyvrtány otvory pro LED diody. Po stranách, v zadní části jsou umístěny LED diody nahrazující směrová světla. Mezi těmito světly se nacházejí „automatická brzdná světla“. Tyto LED diody signalizují, zvýšením intenzity svitu, že dochází k automatickému brzdění. V klidovém režimu nahrazují zadní obrysová světla.

Obrázek 9 Pohled zepředu na model automobilu

Obrázek 8 Pohled zezadu na model automobilu

3. Součástky a komponenty

Autíčko je řízené pomocí Arduino UNO, které je osazené mikrokotrolerem ATmega328. Komunikace s aplikací, běžící na systému Android, je zajištěna přes bluetooth. Řízení motorů je zajišťováno pomocí motor shield, který je nasazen přímo na piny Ardruina. Autíčko je poháněno zadní nápravou pomocí původního DC motorku, který se v modelu nacházel. Řízení směru bylo původně řešeno DC motorkem. Z důvodu nemožnosti přesného nastavení polohy kol, bylo toto řešení nahrazeno modelářským servomotorem. Jako bezpečnostní prvek proti čelnímu nabourání je použit ultrazvukový modul. Blinkry, světla a podsvícení autíčka jsou řešeny pomocí led diod, které jsou zapínány pomocí analogových I/O. Dále je těmito výstupy řízen modul s piezoelektrickým bzučákem, který nahrazuje klakson automobilu. Základní vlastnosti Arduina, jednotlivých shieldů a modulů je popsáno níže.

Arduino Uno

Obrázek 10 Arduino Uno

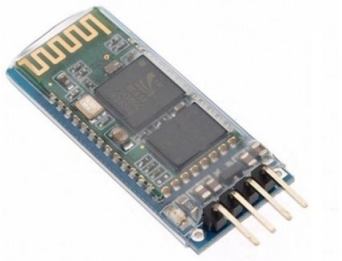
Zdroj: [1]

Mikrokontroler
- Typ: ATmega328
- Flash paměť 32 KB
- SRAM 2 KB
- EEPROM 1 KB

Digitální I/O 14x
- PWM 6 ze 14 DIG. I/O

Analogové I/O 6x

Bluetooth modul HC-05



Obrázek 11 Modul Bluetooth HC-05

Zdroj: [2]

Základní vlastnosti:
- Bluetooth V2.0
- Napájení 3.3-6V
- Komunikace – uart

Ultrazvuk modul HC-SR04

Obrázek 12 Modul ultrazvuku HC-SR04

Zdroj: [3]

Základní vlastnosti:
- Rozsah měření: 5 – 300 cm
- Napájení 5V
- Přesnost 3 mm

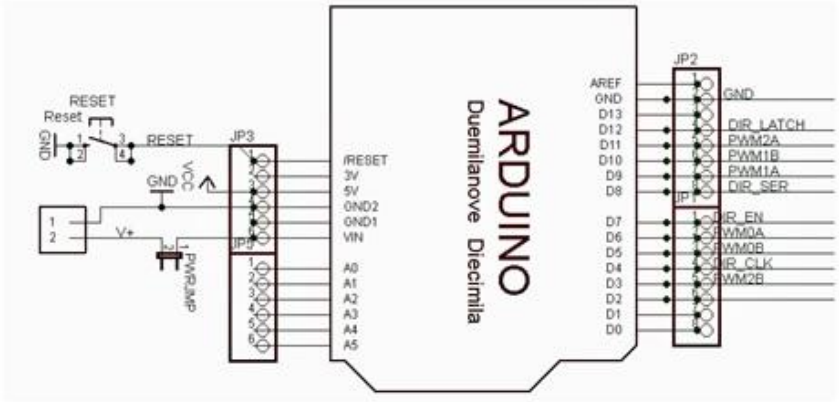
Arduino Motor Servo Shield L293D H-můstek

Obrázek 13 Motor Servo Shield

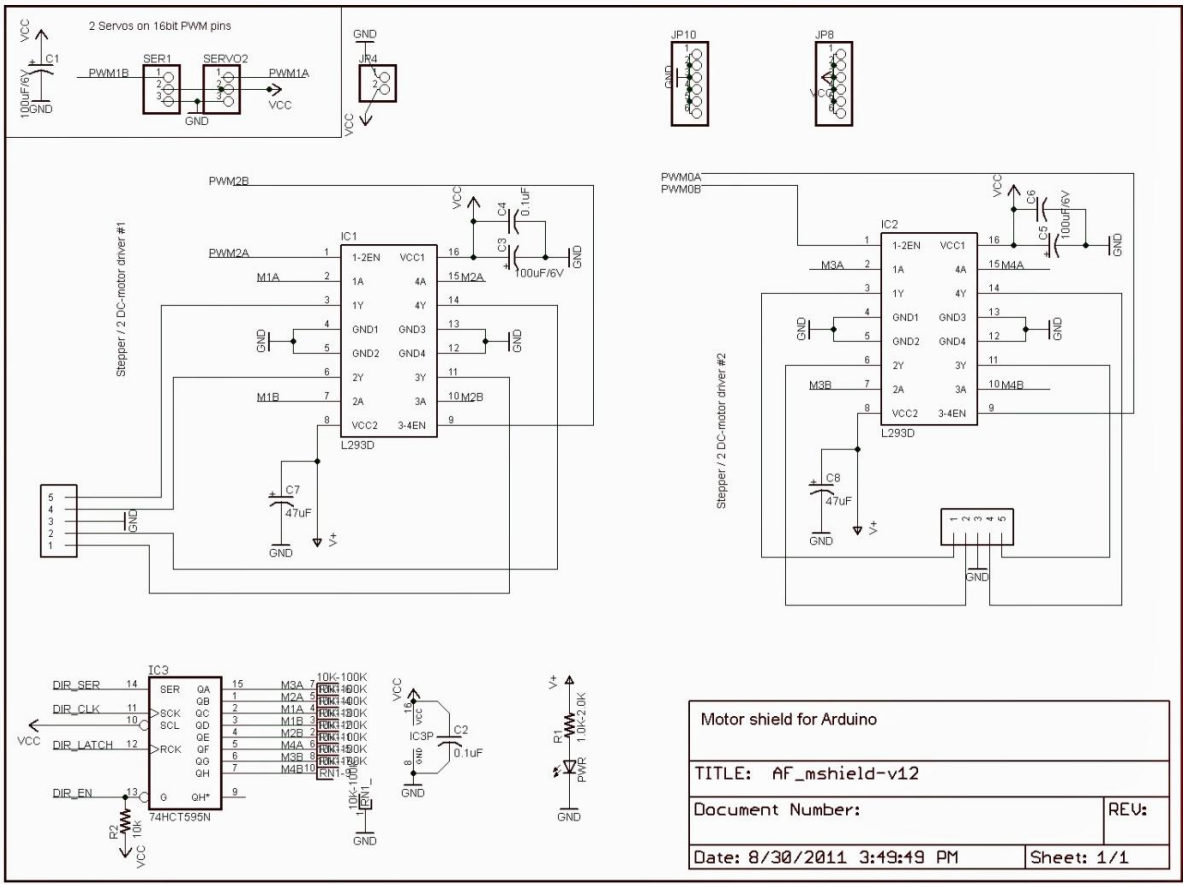
Základní vlastnosti:

- 2x konektor pro servomotor
- 0-4x připojení DC motorku (8-bitové řízení)
- 0-2x krokové motorky
- Konektor pro externí napájení

Vnitřní zapojení:



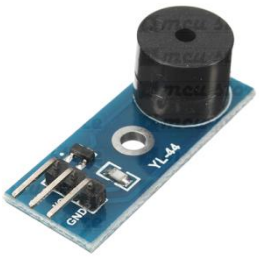
Obrázek 14 Zapojení motor shieldu do arduina



Obrázek 15 Vnitřní zapojení motor shieldu

Zdroj: [4]

Bzučák modul



Obrázek 16 Modul bzučáku

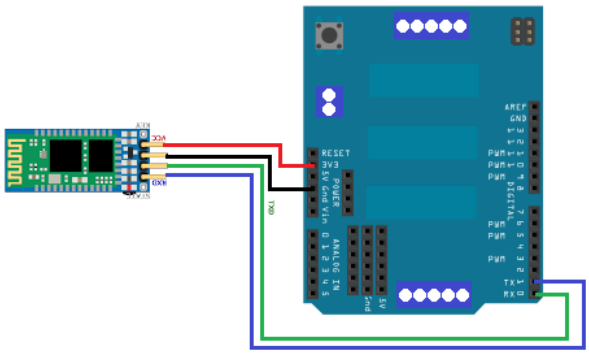
Zdroj: [5]

Základní vlastnosti:

- Aktivní piezo bzučák
- Pracovní napětí 3,3 – 5,0 V
- Spínání tranzistorem 9012

3.1.Zapojení arduino modulů

3.1.1. Bluetooth

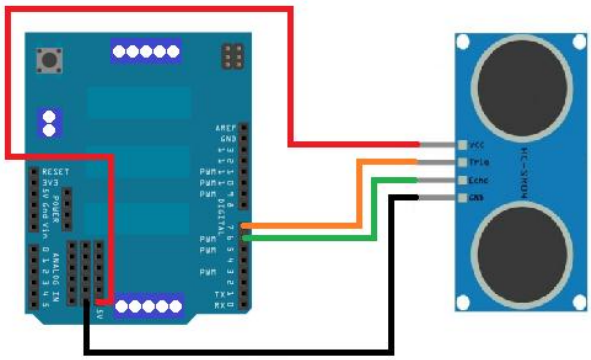


Obrázek 17 Zapojení modulu bluetooth k arduinu

Zapojení	
Arduino	modul
PIN 0 RX	TX*
PIN 1 TX	RX*
3,3V	VCC
GND	GND

* V případě potřeby lze piny TX a RX softwarově přenést například na 2 a 3. Jedná se o zapojení receive do trasmiter a naopak.

3.1.2. Ultrazvuk

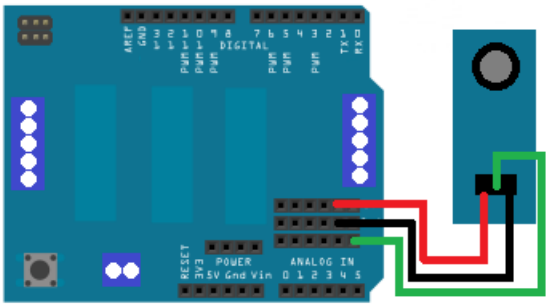


Obrázek 18 Zapojení modulu ultrazvuku k arduinu

Zapojení	
Arduino	**modul**
PIN 6 – dig. pwm	ECHO
PIN 7 – dig.	TRIG
5V	VCC
GND	GND

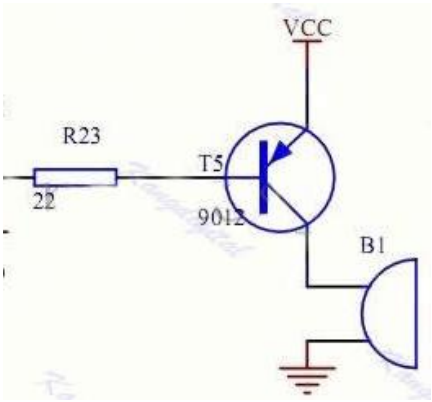
- Trig – „vstup“ do ultrazvuku z arduina, který vyšle ultrazvukovou vlnu.
- Echo – „výstup“ z ultrazvuku do arduina, který vyšle informaci o příchozím signálu

3.1.3. Bzučák



Zapojení	
Arduino	**modul**
PIN A5 - analog	I/O
5V	VCC
GND	GND

Obrázek 19 Zapojení modulu piezo bzučáku k arduinu



Obrázek 20 Schéma zapojení piezo bzučáku.

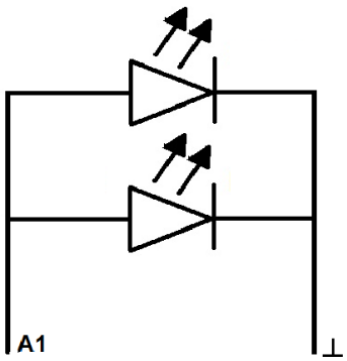
- I/O – vstup zapojený přes odpor R23. Tento vstup otvírá tranzistor T5, při 0V a dojde tím k pískání. Při hodnotě 5V na tomto vstupu je tranzistor T5 zavřený a nedochází k pískání (obr.20).

Zdroj: [5]

4. Zapojení arduina s ostatními součástkami

V této části dokumentace se nachází přehled zapojení použitých elektrotechnických součástek, které jsme využili pro realizaci autentických prvků.

4.1.1. Zapojení podsvícení podvozku

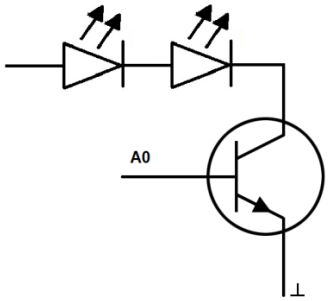


Zapojení je realizováno jako paralelní zapojení led diod zelené barvy. Obvod je zapnut při LOG 1 na analogovém pinu A1. Při sepnutí je obvod napájen 5V.

LED 3mm zelená; 30°; 6000 mcd; 3,3 V; 20 mA
Průhlednost pouzdra: čirá
Barva pouzdra: čirá

Obrázek 21 Zapojení podsvícení

4.1.2. Zapojení předních světel



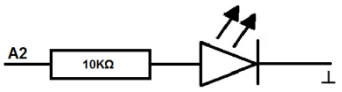
Obrázek 22 Zapojení světlometů

Zapojení (Obrázek 22) je realizováno jako sériové zapojení led diod bílé barvy. Obvod je zapnut při LOG 1 na analogovém pinu A0, který je přiveden na tranzistor typu NPN (KC635). Při sepnutí je obvod napájen plným napětím z baterií (7,5 V).

LED 3mm bílá; 23-30°; 3000-10000 mcd; 3,2 V; 20 mA
Průhlednost pouzdra: čirá
Barva pouzdra: čirá

4.1.3. Zapojení led diody – couvací

Zapojení (Obrázek 23) je realizováno jako sériové zapojení rezistoru (10kΩ) a led diod bílé barvy. Obvod je zapnut při LOG 1 na analogovém pinu A2. Při sepnutí je obvod napájen napětím (5 V).

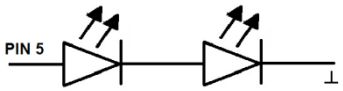


Obrázek 23 Zapojení couvací LED D.

LED 3mm bílá; 23-30°; 3000-10000 mcd; 3,2 V; 20 mA
Průhlednost pouzdra: čirá
Barva pouzdra: čirá

4.1.4. Zapojení led diod – automatického brzdění

Zapojení (Obrázek 24) je realizováno jako sériové zapojení led diod červené barvy. Obvod je zapnut při LOG 1 na digitálním pinu 5. Při zapnutí napájení automobilu, je tento obvod napájen 2,5 V. V této situaci plní funkci zadních obrysových světel. Dále jsou používány jako informace pro uživatele, že dochází k automatickému brzdnému systému. Při této funkci je obvod napájen 5V. Brzdný systém je popsán v kapitole 5.1.4.

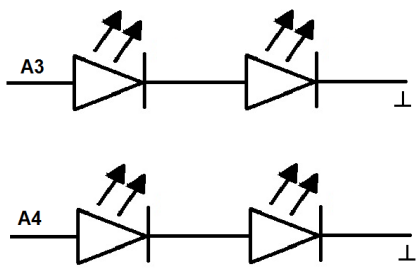


Obrázek 24 Zapojení LED d. automatického brždění

LED 3mm červená; 60°; 1,3-5 mcd; 2 V; 20 mA
Průhlednost pouzdra: difúzní
Barva pouzdra: červená

4.1.5. Zapojení led diod – blinkrů

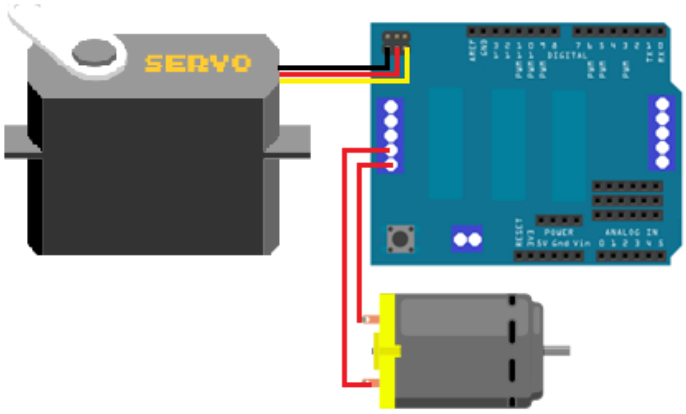
Zapojení (Obrázek 25) je realizováno jako sériové zapojení led diod oranžové barvy. Obvod je zapnut při LOG 1 na analogovém pinu A3, plní funkci směrových světel na levé straně automobilu. Druhý obvod je zapnut při LOG 1 na analogovém pinu A4, plní funkci směrových světel na pravé straně automobilu. Oba obvody jsou napájeny 5 V.



Obrázek 25 Zapojení LED diod blinkrů

LED 3mm oranžová; 60°; 8-20 mcd; 2 V; 20 mA
Průhlednost pouzdra: difuzní
Barva pouzdra: oranžová

4.1.6. Zapojení motorů (servomotor, DC motor)



Na Obrázek 26 je vyobrazeno zapojení servomotoru a DC motorku k motor shieldu. Servomotor se připojuje pomocí třech vodičů. Červený a černý vodič slouží pro napájení. Žlutý vodič slouží jako signální pro nastavení správné pozice. DC motorek je napojen do svorkovnice, která je označena jako M2.

Obrázek 26 Zapojení motorů

4.1.7. Přehled zapojení arduino modulů a ostatních součástek

Název modulu/shieldu	Arduino UNO	Shield/modul
Motor shield	Shield se nasazuje na arduino UNO	

Bluetooth	PIN 0 RX	TX
	PIN 1 TX	RX
	3,3V	VCC
	GND	GND

Ultrazvuk	PIN 6 – dig. pwm	ECHO
	PIN 7 – dig.	TRIG
	5V	VCC
	GND	GND

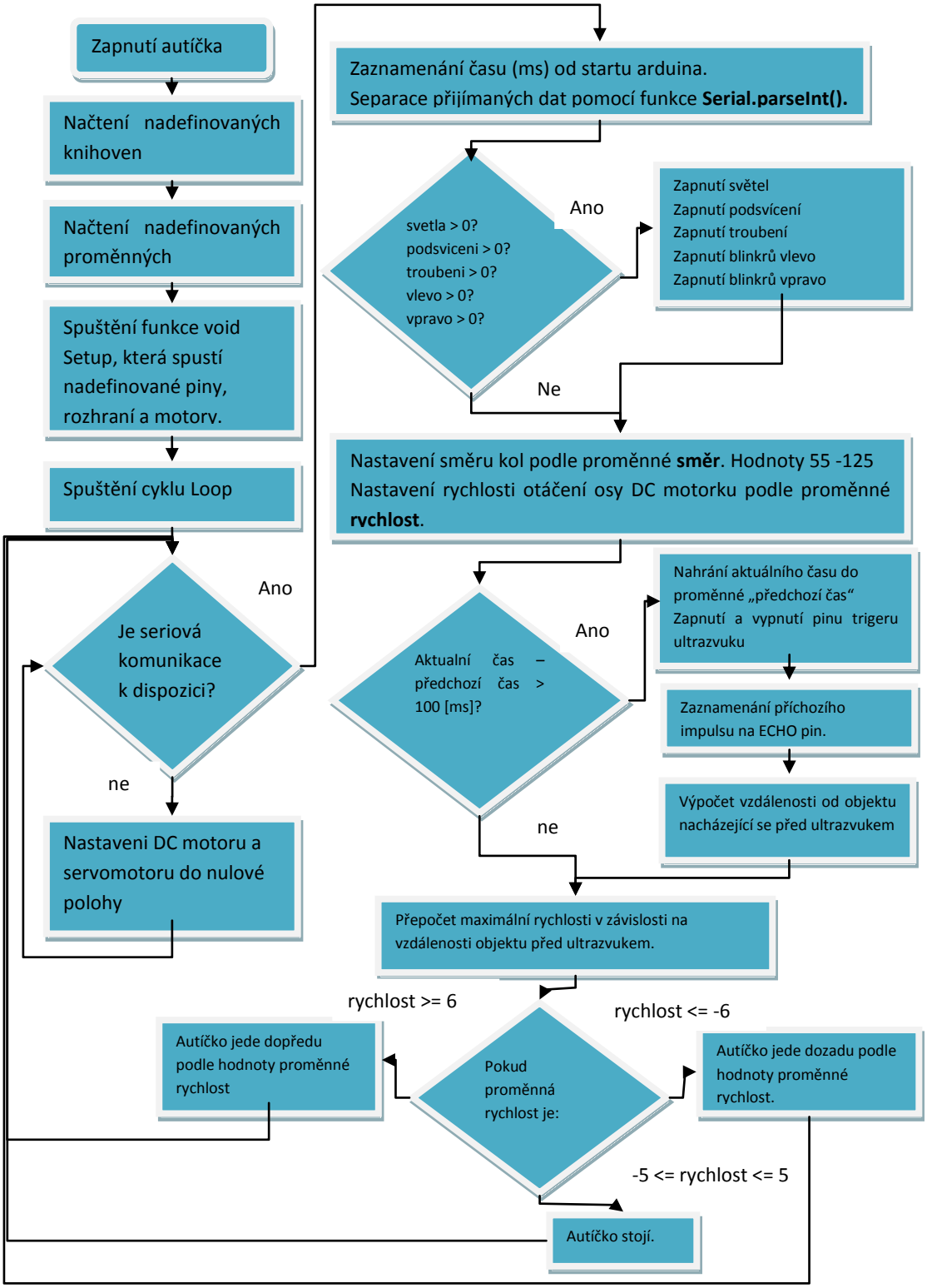
Piezo bzučák	PIN A5 - analog	I/O
	5V	VCC
	GND	GND

Ostatní součástky	Arduino UNO
LED diody – Podsv. podvozku	A1 – analog. I/O
LED diody – přední světlomety	A0 – analog. I/O
LED dioda – couvací	A2 – analog. I/O
LED diody – autom. brždění	5 – digitální I/O
LED diody – levá směrová sv.	A3 – analog. I/O
LED diody – pravá směrová sv.	A4 – analog. I/O

4.1.8. Cenová kalkulace

Název	Počet	Cena
Arduino UNO	1x	379 Kč
Motor shield	1x	169 Kč
Bluetooth	1x	229 Kč
Ultrazvuk	1x	69 Kč
Piezo bzučák	1x	38 Kč
Model autíčka + poštovné		90 Kč
Servomotor pro ovládání předních kol		150 Kč
Ostatní (LED diody, cín, dráty, atd…)		50 Kč
Cena celkem		**1174 Kč**

5. Arduino – zdrojový kód



Obrázek 27 Obecný vývojový diagram

Tento vývojový diagram (Obrázek 27) popisuje obecné fungování kódu v Arduinu.

5.1.1. Knihovny

```
#include <Servo.h>
#include <AFMotor.h>
```

Příkaz **#include** si propůjčuje části kódů z externích souborů.

Knihovna **servo.h** je knihovna, se kterou je možné ovládat servomotor.
Knihovna **AFMotor.h** je knihovna, se kterou je možné ovládat servomotor.

Knihovny ke stažení

1. **servo.h** – je nainstalována společně s vývojovým prostředím Arduino
2. **AFmotor.h** - https://github.com/adafruit/Adafruit-Motor-Shield-library

5.1.2. Definování proměnných

```
#define ECHOPIN 6
#define TRIGPIN 7
```

#define – nahrazuje číslo (6) na textovou formu (ECHOPIN), aby zdrojový kód byl přehlednější a logičtější.

AF_DCMotor motor(2); - Vytvoření objektu pro práci s DC motorkem s názvem motor.
Servo myservo; - Vytvoření objektu pro práci se servem s názvem myservo.

```
int rychlost;
int smer;
int troubeni;
int vlevo;
int vpravo;
int svetla;
int podsviceni;
int rychlost_max;
```

int – typ **integer** – tyto proměnné nabývají pouze celých čísel

```
float distance;
float vzdal_posuzovana;
```

float – typ **float** – tyto proměnné nabývají pouze čísel, které mají plovoucí desetinou čárku.

```
unsigned long aktualniMillis;
unsigned long predchoziMillis;
```

unsigned long – Jedná se o celé číslo bez znaménka. Má rozsah od 0 do 4294967295.

5.1.3. Funkce SETUP

void setup()
{

Po zapnutí hlavního přepínače autíčka do polohy ON. Proběhne tato funkce pouze jednou.

Serial.begin(9600);

- Zapnutí seriové komunikace

myservo.attach(9);
myservo.write(90);

- Nastavení servomotoru.
  - **myservo.attach(9)** – oživení servomotoru na pinu 9
  - **myservo.write(90)** – nastavení pozice servomotoru na pozici 90, která nastaví kola do výchozí polohy

motor.setSpeed(0);
motor.run(RELEASE);

- Nastavení DC motoru.
  - **motor.run(RELEASE)** – oživení DC motorku
  - **motor.setSpeed(0)** – nastavení rychlosti otáčení na nulové otáčky.

pinMode(ECHOPIN, INPUT);
pinMode(TRIGPIN, OUTPUT);
pinMode(A5,OUTPUT);
pinMode(A4,OUTPUT);
pinMode(A3,OUTPUT);
pinMode(A2,OUTPUT);
pinMode(A1,OUTPUT);
pinMode(A0,OUTPUT);
pinMode(5,OUTPUT);

- **pinMode**(označení pinu, nastavení pinu (vstupní/výstupní))

analogWrite(A5,255) ;
analogWrite(5,25);
}

- **analogWrite**(označení pinu, nastavení voltáže 0-5V hodnotami 0 – 255)
- Tento příkaz se využívá hlavně u **digitálních** pinů, které mají **PWM**. Lze dosáhnout přesného nastavení napětí, které uživatel požaduje.
- U **analogových** pinů lze nastavit jen hraniční hodnoty a to 0V nebo 5V

5.1.4. Cyklus loop

void loop() {

Tato funkce probíhá opakovaně do doby vypnutí hlavního přepínače autíčka do polohy OFF.

while (Serial.available()){

Tento cyklus probíhá opakovaně, dokud je sériová komunikace k dispozici.

aktualniMillis = millis();

millis() – Zaznamenání času (ms) od startu arduina do proměnné **aktualniMillis**.

rychlost = Serial.parseInt();
smer = Serial.parseInt();
troubeni = Serial.parseInt();
vpravo = Serial.parseInt();
vlevo = Serial.parseInt();
svetla = Serial.parseInt();
podsviceni = Serial.parseInt();

Serial.parseInt() – funkce která z přijatého řetězce vyseparuje jen potřebná celá čísla.
Hodnoty Integer jsou vždy oddělené nějakou non Integer hodnotou, v našem případě se jedná například o „x“,“y“,“t“,“k“.

Aby došlo ke správnému přiřazení vyseparovaných celých čísel, je nutné příslušné proměnné seřadit tak, jak přicházejí v textovém řetězci.

if (svetla > 0) analogWrite(A0,255); else analogWrite(A0,0);
if (podsviceni > 0) analogWrite(A1,255); else analogWrite(A1,0);
if (vpravo > 0) analogWrite(A4,255); else analogWrite(A4,0) ;
if (vlevo > 0) analogWrite(A3,255); else analogWrite(A3,0);

Tyto podmínky se dotazují, zda proměnná nabývá hodnot větších jak 0. Pokud ano, nastaví příslušný analogový výstup na hladinu 255 (5V). Pokud je hodnota rovna 0, je nastavena na příslušných analogových výstupech hladina 0 (0V).

if (troubeni > 0) analogWrite(A5,0); else analogWrite(A5,255);

Pro splnění podmínky troubení musí nabývat proměnná hodnot větších jak 0. Pokud tomu tak je, dojde k nastavení hladiny na 0 (0V). Jestliže tomu tak není, je na analogovém pinu A5 nadále nastavena hladina 255 (5V). Toto nastavení vychází, ze zapojení piezo bzučáku, který je ovládaný pomocí PNP tranzistoru.

```
smer = constrain(smer, 55, 125);
myservo.write(smer);
```

Tyto dva řádky kódu nastavují polohu servomotoru, který ovládá směr předních kol autíčka.

První řádek omezuje hodnotu v proměnné **smer**, podle krajních hodnot možného natočení předních kol (55 – 125). Tato upravená hodnota se zapíše zpět do proměnné **smer**. Druhý řádek kódu nastavuje servomotor do příslušné polohy podle hodnoty nacházející se v proměnné **smer**.

```
if(aktualniMillis - predchoziMillis > 100) { //zaslat signál - 100ms od poslední akce?
        predchoziMillis = aktualniMillis; //v tento čas jsem provedl poslední akci
```

Touto podmínkou je zajištěno, že dochází k aktivaci ultrazvuku jednou za 100ms.

Po splnění této podmínky je do proměnné predchoziMillis vypsána hodnota z proměnné aktualniMillis, která obsahuje hodnotu, která je zaznamenávána na začátku while cyklu.

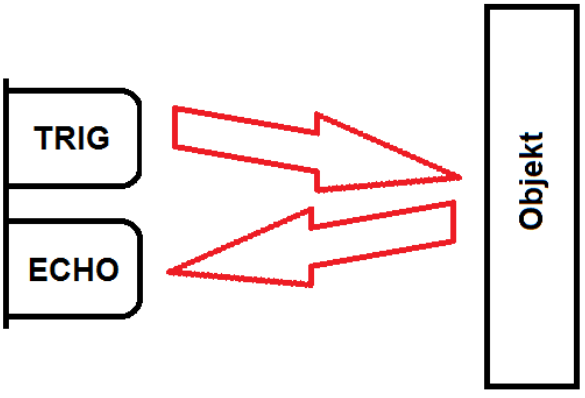
```
digitalWrite(TRIGPIN, LOW);
delayMicroseconds(2);
digitalWrite(TRIGPIN, HIGH);
delayMicroseconds(10);
digitalWrite(TRIGPIN, LOW);
```

Touto částí kódu je softwarově zapínán a vypínán pin, na který je připojen TRIGER ultrazvuku a při aktivaci zasílá ultrazvukový impuls.

```
distance = pulseIn(ECHOPIN, HIGH);
distance= distance*0.017315;
```

}

pulseIn(ECHOPIN, HIGH). Tato funkce vrací počet mikrosekund od detekování pulsu v hodnotě HIGH. Po vyslání impulsu do modulu program počká na zpětné odeslání pulzu od modulu, kdy funkce pulseIn() nám vrátí potřebný počet mikrosekund. S tímto číslem můžeme dále pracovat. Vezmeme v úvahu rychlost zvuku 346,3 m*s-1 a to při teplotě suchého vzduchu 25°C. To znamená, že za 1 mikrosekundu urazí v metrech 346,3/1000000, což je 0,0003463 metru. Převedeno na cm to je 0,03463cm/mikrosekundu. Vzhledem k tomu, že signál jde od čidla k

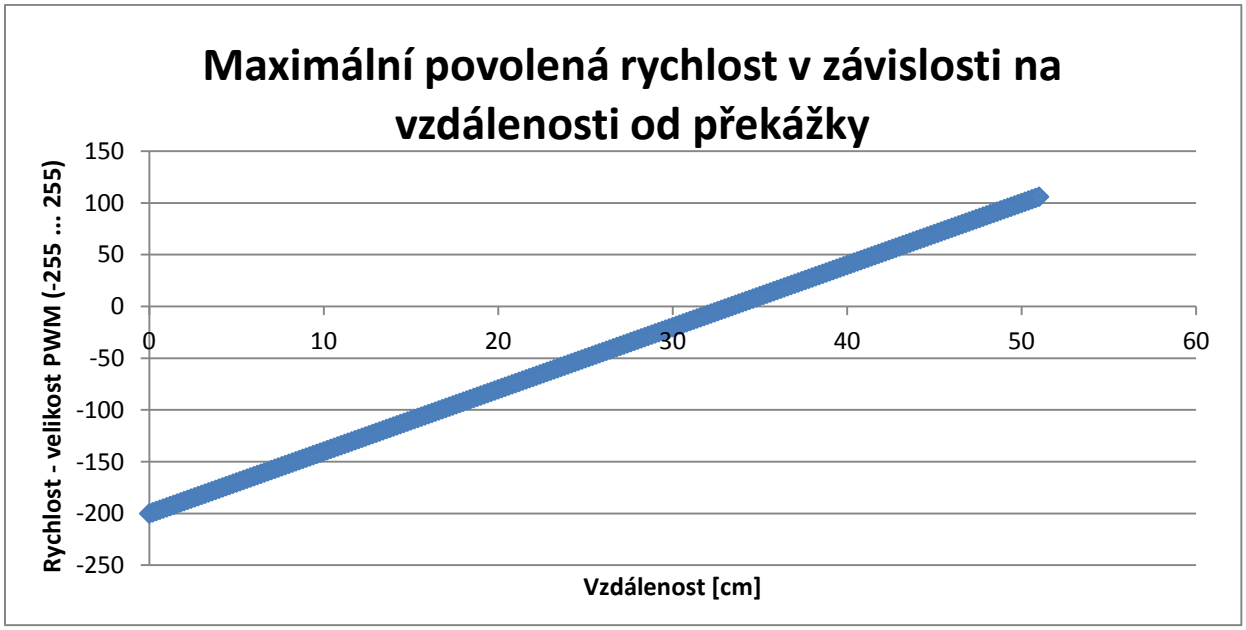


Obrázek 28 Princip měření ultrazvukem

předmětu, kde se odrazí a zase zpět, musíme tuto vzdálenost ještě vydělit číslem 2 (viz Obrázek 28). Výsledek je, že se vzdálenost bude rovnat počtem mikrosekund násobených číslem 0,017315. Výsledek je zaznamenán do proměnné distance. [6]

Další část kódu zajišťuje plynulé zpomalení autíčka ve chvíli, kdy by mělo narazit do překážky nebo by se pohybovalo v její blízkosti. Algoritmus neumožní autíčku v blízkosti překážky překračovat maximální povolenou rychlost a tak se rozjet natolik, aby autíčko narazilo. Křivka (viz obr. 28.) je nastavena tak, aby ani při přímém rozjetí nehrozil velký nebo pokud možno žádný náraz (to ovlivňuje zejména povrch (a jeho koeficient valivého tření s koly autíčka), po kterém se autíčko pohybuje). Tato část kódu ovlivňuje rychlost pohybu autíčka – už tedy nejde o rychlost, kterou poslala aplikace v mobilním telefonu. Je tedy možné říci, že ve vzdálenosti pod 50 cm od překážky získává software autíčka „nadvládu“ nad jeho rychlostí.

```
if (rychlost > 0){ vzdal_posuzovana = rychlost / 5;
if (vzdal_posuzovana > distance){ rychlost_max = round((distance*6)-200);
if (rychlost_max < rychlost){rychlost = rychlost_max;
analogWrite(5,255);
}}else analogWrite(5,25);}
if (distance < 15) rychlost = -99;
```

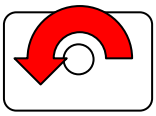


Obrázek 29. Graf maximální povolené rychlosti v závislosti na vzdálenosti překážky

```
if (rychlost >= 6) {
motor.run(FORWARD);
motor.setSpeed(rychlost);
analogWrite(A2,0); }
```

Pokud je rychlost větší nebo rovna šesti jsou výstupní svorky na motor shieldu nastaveny tak, aby se osa motoru otáčela doleva (Obrázek 30). Autíčko se začne pohybovat směrem dopředu. Přijímané

hodnoty se pohybují od 6 do 255, kde 255 je nejvyšší rychlost. Pin A2 na kterém je zapojena LED dioda, imitující couvací světlo je nastaven na 0V.



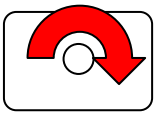
Obrázek 30 Otáčení osy DC motoru

```
if ((rychlost <= 5) && (rychlost >= -5)){
analogWrite(A2,0);
motor.setSpeed(0);
}
```

Pokud je hodnota rychlosti v rozsahu od mínus pěti do pěti tak je motor nastaven na 0. Tato podmínka odstraňuje lehké nechtěné náklony telefonu dopředu a dozadu. Dále nastavuje pin A2 na hodnotu 0V.

```
if (rychlost <= -6) {
motor.run(BACKWARD);
motor.setSpeed(-200-(rychlost*2));
analogWrite(A2,255);
}
}
```

Pokud je rychlost menší nebo rovna mínus šesti jsou výstupní svorky na motor shieldu nastaveny tak, aby se osa motoru otáčela doprava a tím autíčko začalo couvat. Přijímané hodnoty se pohybují od -6 do -255, kde -6 je nejvyšší rychlost. Proto jsou příchozí hodnoty invertovány, aby došlo ke správnému zrychlování otáček motoru. Pin A2 je nastaven na 5V a tím signalizuje, že autíčko couvá.



Obrázek 31 Otáčení osy DC motoru

```
  motor.setSpeed(0);
  motor.run(RELEASE);
  myservo.write(90);
}
```

Tyto příkazy proběhnout tehdy pokud dojde k ukončení spojení mezi autíčkem a telefonem. DC motorek nastaví hodnotu na 0 a autíčko se zastaví. Dále je servomotor nastaven na 90 a tím jsou kola srovnána do výchozí polohy.

6. Dokumentace k mobilní aplikaci

Mobilní aplikace pro ovládání autíčka je vytvořena v prostředí App Inventor 2. Toto cloudové vývojové prostředí umožňuje zadarmo sestavovat vlastní aplikaci z předprogramovaných bloků a znatelně tak usnadňuje prvotní vývoj aplikací pro nezkušené programátory. App Inventor je samozřejmě určen pro programování aplikací na OS Android. Bylo zvažováno i použití prostředí Java Eclipce, ale vzhledem k pravděpodobně velké časové náročnosti byla nakonec tato varianta zamítnuta.

6.1.Cíl aplikace

Cílem aplikace bylo umožnit dálkové ovládání autíčka přes tzv. „chytrý mobilní telefon“ s bluetooth. Sekundárním cílem bylo autíčko ovládat na základě polohy akcelerometru umístěném v mobilním telefonu a případně, aby na displeji telefonu byla zobrazena grafika momentálně odesílaných dat (jakási předběžná zpětná vazba). Dále se v průběhu práce na autíčku objevily další vhodné ovládací možnosti, tak je z mobilní aplikace možné ovládat klakson, pravé/levé/výstražná blikající světla, tempomat, podsvícení a světla.

6.2.Jednoduchý úvod do programování v App Inventoru 2

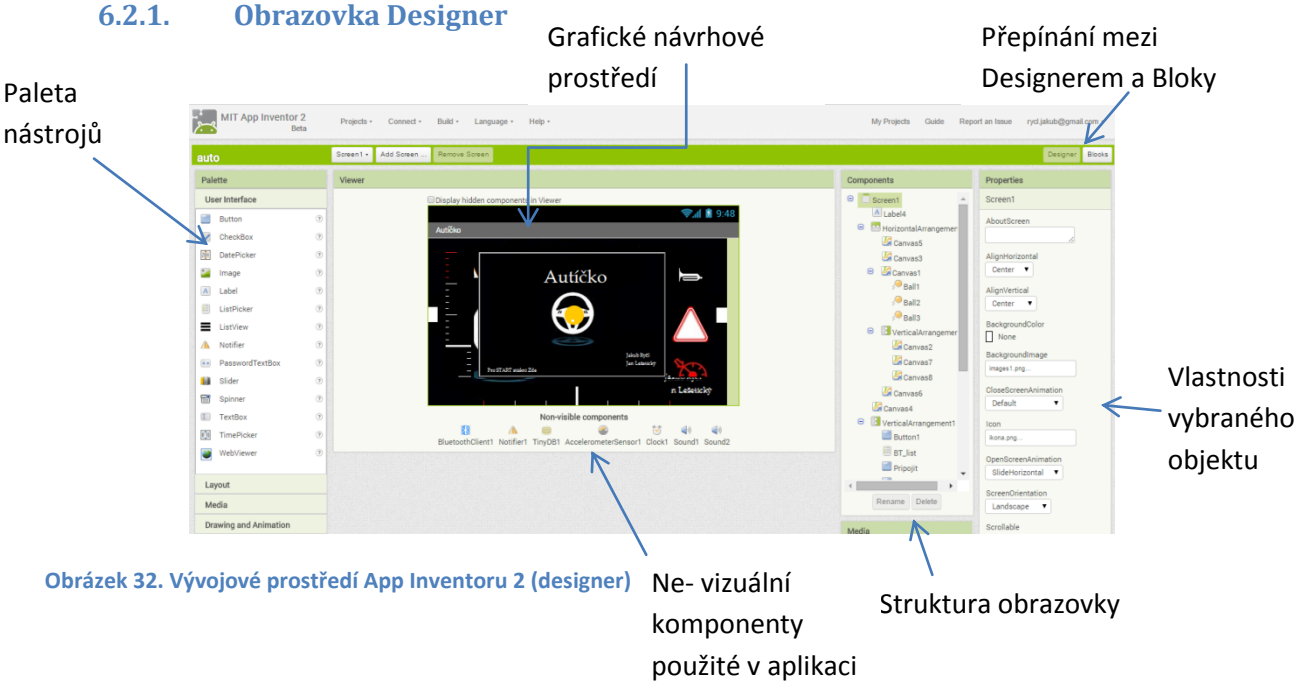
Pro pochopení dokumentace je vhodné zmínit, jakým způsobem tento ne zcela standardní nástroj funguje.

Vývojové prostředí má dvě základní obrazovky, na kterých se dále pracuje, jsou to obrazovky:

- Designer,
- Blocks.

V obrazovce „Designer“ se rozmisťují jednotlivé prvky na obrazovce a rozhoduje se o jejich grafických a inicializačních vlastnostech, naproti to tomu v obrazovce „Blocks“ se pomocí jednotlivých bloků sestavuje program. Blokem je v tomto pojetí každý jednotlivý prvek programovacího jazyka – tedy jak funkce, tak například podmínka, každá jednotlivá hodnota, deklarace atd. (v podstatě tedy cokoliv).

6.2.1. Obrazovka Designer



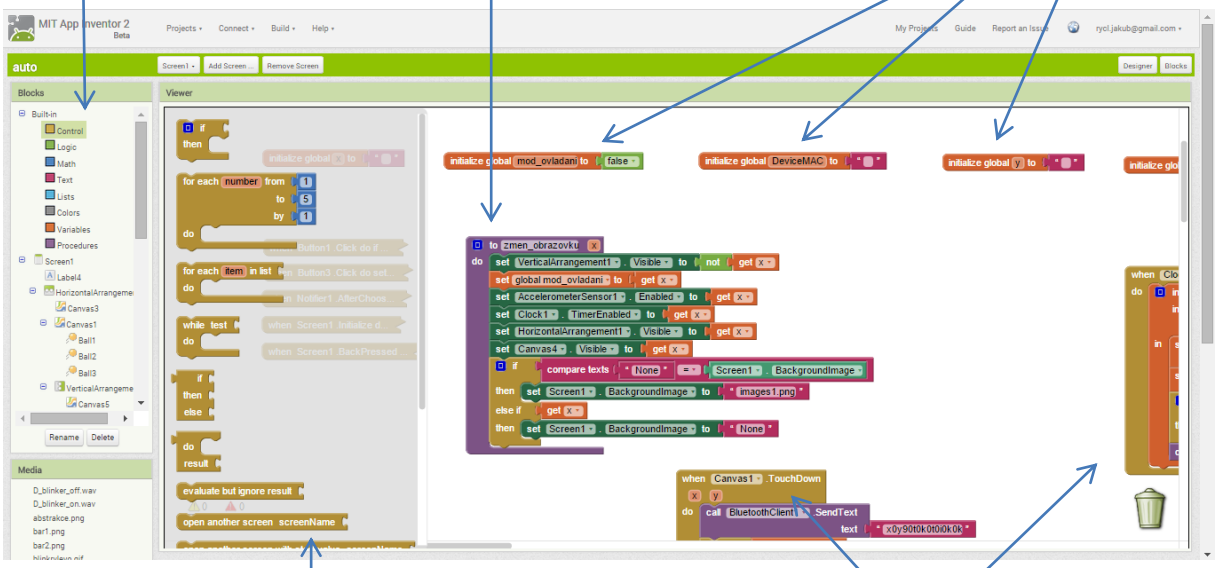
Obrázek 32. Vývojové prostředí App Inventoru 2 (designer)

6.2.2. Obrazovka Block

Jednotlivé funkce roztříděné do kategorií

Blok vlastní funkce

Deklarované proměnné



Obrázek 33 Vývojové prostředí App Inventoru 2 (obrazovka block)

Výběr funkcí z kategorie „Control“

Blok Eventu

Programuje se na základě událostí, jednotlivé události neboli Eventy mají vždy své vlastní bloky a pro každou jedinečnou událost musí být daný blok také jedinečný. Bloky se nedají navzájem sdružovat ani jinak kombinovat. Jednoduchým příkladem Eventu může být například stisk tlačítka.

Umístění bloků na Obrazovce „Blocks“ nehraje žádnou roli, což je jedna z věcí, na které si člověk navyklý na strukturované programovací jazyky, musí zvykat. Existují zde různé možnosti automatického zarovnání, ale přehlednosti moc nepřidají, proto je lepší vytvořit si svůj vlastní systém pro snadnou orientaci.

App inventor má i řadu dalších nevýhod, mezi nejzásadnější patří již zmíněné cloudové řešení, které nejenže vyžaduje internetové připojení, ale vyžaduje ho i v relativně vysoké kvalitě (v případě přerušení spojení se totiž může s projektem stát v podstatě cokoliv a člověk se při jeho opětovném navázání občas nestačí divit, jaké změny aplikace sama od sebe vytvořila).

6.3.Dokumentace vlastní aplikace „Autíčko“

Dokumentace se skládá z koncepce a postupného popisu jednotlivých komponent aplikace.

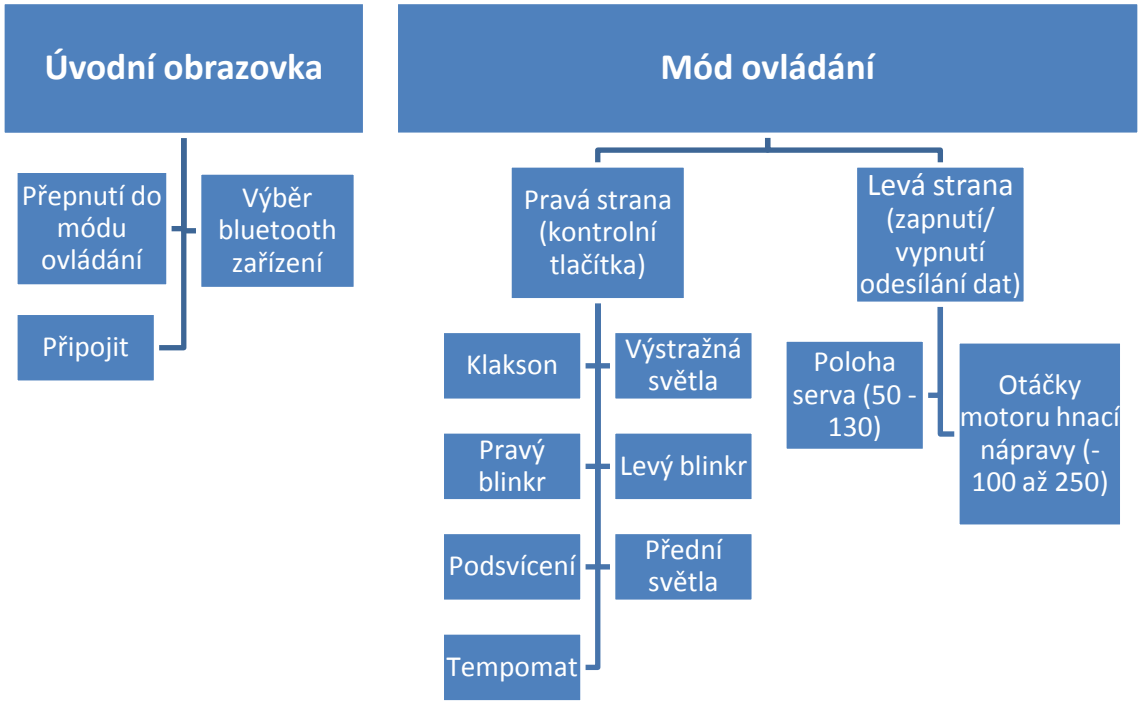
6.3.1. Koncepce aplikace

Koncepce je postavena na dvou „obrazovkách“, kde na první je nastaveno připojení k Bluethootu (dále BT) a na druhé je samotné ovládání autíčka.

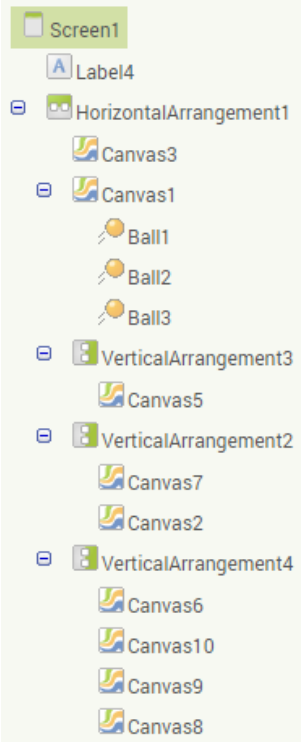
I přesto, že má aplikace dvě obrazovky je postavena pouze na jednom takzvaném Screenu, to je důležité převážně z důvodu sdílení dat mezi jednotlivými screeny. V našem případě by bylo vhodné mít dva screeny na jednom mít nastavení a na druhém ovládání samotné, zde ovšem nastává

problém se sdílením dat při navázání spojení pomocí BT. Z druhého screenu je velice obtížné či možná přímo nemožné přistupovat k objektu (BT) na 1. screenu, kde je navázaná komunikace. Tento problém je tedy „obcházen“ použitím pouze jednoho Screenu a skrýváním momentálně nepoužívaných komponent - nastavením visibility(viditelnosti).

6.3.2. Ovládací možnosti – struktura

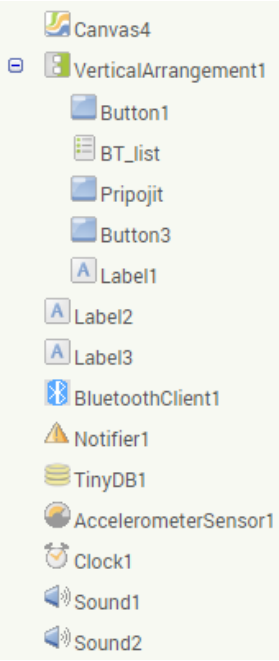


6.3.3. Komponenty použité v aplikaci



- Jediná obrazovka použitá v aplikaci
- Popisek používaný pro testování (zobrazuje odesílaná data)
- Blok pro horizontální uspořádání
- Zobrazuje momentální rychlost
- Hlavní ovládací prvek
- Zobrazuje momentální náklon telefonu
- Zobrazuje střed
- Zobrazuje se při neaktivitě jako šedý střed
- Blok pro vertikální uspořádání
- Blinkr vlevo
- Blok pro vertikální uspořádání
- Výstražná světla
- Klakson
- Blok pro vertikální uspořádání
- Blinkr vpravo
- Světla
- Podsvícení
- Tempomat

Obrázek 34 Komponenty použité v aplikaci (část 1)



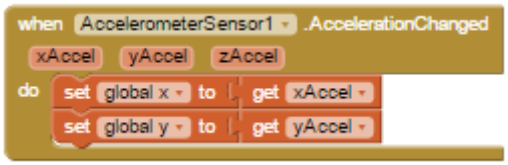
- Obrázek natočení mobilního tel. v horizontálním směru
- Blok pro vertikální uspořádání
- Přepnout na mód ovládání
- „Tlačítko“ výběr BT (jedná se o ListBox)
- Tlačítko propojit BT
- „Skryté“ tlačítko pro testování aplikace
- Skrytý popisek využívaný jako proměnná - zda je BT připojeno
- Popisek
- Popisek
- BT Client1
- Zobrazování zpráv (jak výstražných tak dialogových)
- Databáze (spárovaných zařízení)
- Akcelerometr
- Časovač
- Přehrávání zvuku 1
- Přehrávání zvuku 2

Obrázek 35 Komponenty použité v aplikaci (část 2)

6.4.Princip ovládání

Ovládání je postaveno na jednotlivých událostech, které se vzájemně doplňují.

6.4.1. Událost AccelerometerSensor1.AccelerationChanged



Obrázek 36 Událost AccelerometerSensor1.AccelerationChanged

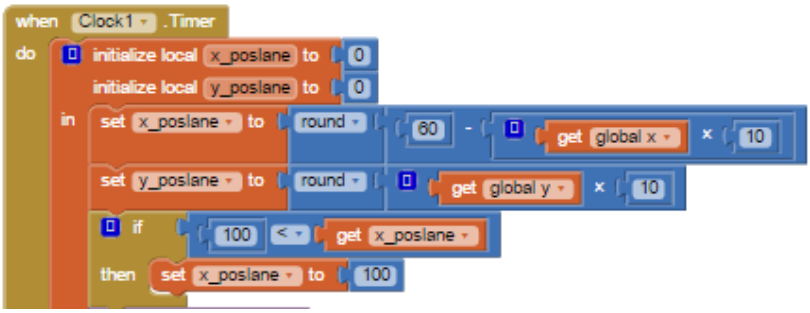
Tato událost pracuje s akcelerometrem integrovaným v mobilním telefonu a předává hodnoty momentálního natočení telefonu ve třech osách v rozsahu od -10 do 10. Jak plyne z názvu, událost je provedena při každé změně jednotlivých hodnot. Pro ovládání autíčka je nutné znát natočení alespoň ve dvou osách (pro pohyb dopředu/dozadu a vpravo/vlevo), z důvodu jednoznačného určení o které osy se jedná, bylo stanoveno stále natočení obrazovky aplikace – a to jako vlastnost Screen1.Screenorientation – Landscape. V samotné vnitřní části bloku se hodnoty pro další použití v programu ukládají do globálních proměnných x a y.

Podstatné je, aby v této události byl co nejméně náročný kód, protože se opakuje velmi často a proto i při mírné výpočetní náročnosti dokáže aplikaci velmi zpomalit.

6.4.2. Událost Timer1.Clock

Všechny hodnoty (hodnoty jednotlivých funkcí) jsou odesílány jednou za 100 ms. Ústřední roli tedy hraje událost (event) Timer1.Clock a jedna z jejích funkcí Bluetoothclient1.SendText. Je třeba říci, že zdokumentovat událost Timer1.Clock je vzhledem k její rozsáhlosti velmi složité (provádí se mnoho instrukcí – program uvádí, že je v události přítomno 247 bloků). Vše co je v události se provede každých 100ms, a proto je zde i ovládání veškerých grafických funkcí.

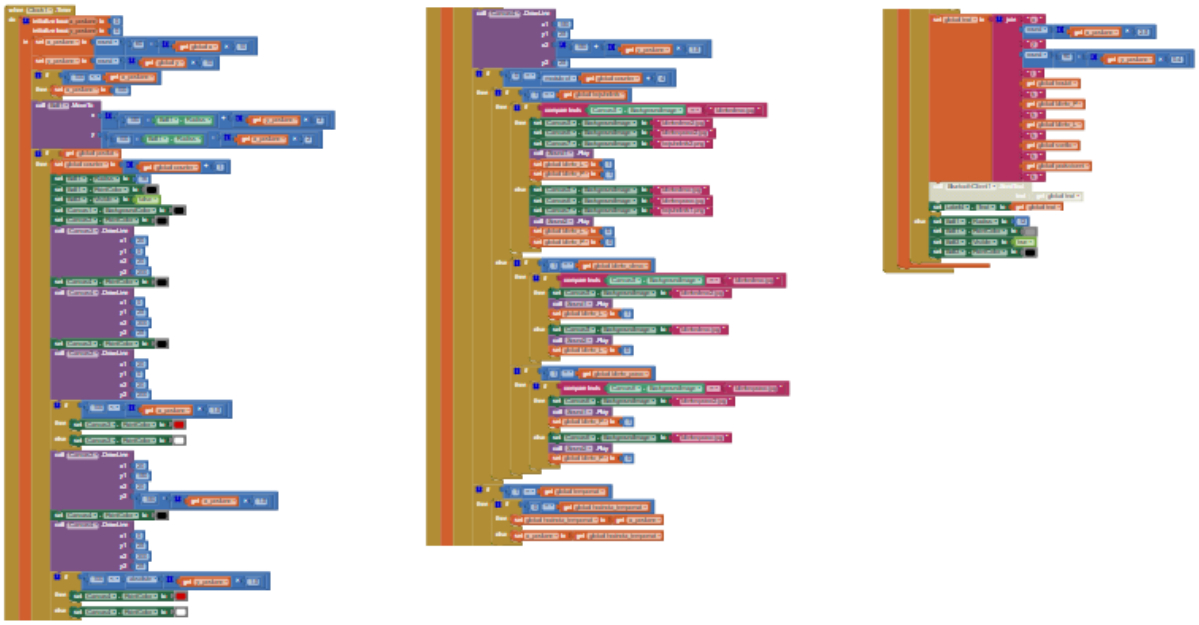
V události jsou deklarovány vnitřní proměnné x_poslane a y_poslane, které slouží pro práci s globálními proměnnými x a y.



Obrázek 37. Úvodní bloky Události Clock1.Timer

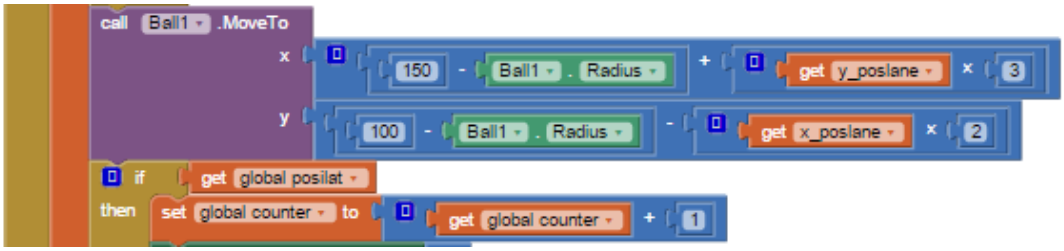
Do proměnné x_poslane se ukládá hodnota pro natočení dopředu/dozadu – aby se docílilo správného rozsahu (např. pro grafické funkce atd.), násobí se hodnota global x (rozsah -10-10) desetkrát, rozsah se tedy zvýší na -100 – 100 a vzhledem k tomu, že je vhodné pro pohodlné ovládání mít nulovou pozici posunutou, je vhodné odečítat tuto hodnotu od šedesáti, rozsah je pak tedy (-40 až 160) pro hodnoty nad 100 by však uživatel musel nepříjemně „překrucovat“ telefon a tak veškeré hodnoty nad tuto mez jsou brány pouze jako 100 – výsledný rozsah je tedy (-40 až 100) s tím, že nulová pozice je posunuta oproti hardwarově výchozí tak, aby byla pro uživatele co nejpřirozenější. Nevýhoda zde nastává pouze jedna a to, že pro pohyb vzad je následně rozsah menší a uživatel nemá v žádné pozici možnost jet dozadu rychleji než cca polovinou maximální možné rychlosti.

V proměnné y_poslane je ukládán desetkrát zvětšený zaokrouhlený rozsah proměnné global y.



Obrázek 38. Celá událost Clock1.Timer

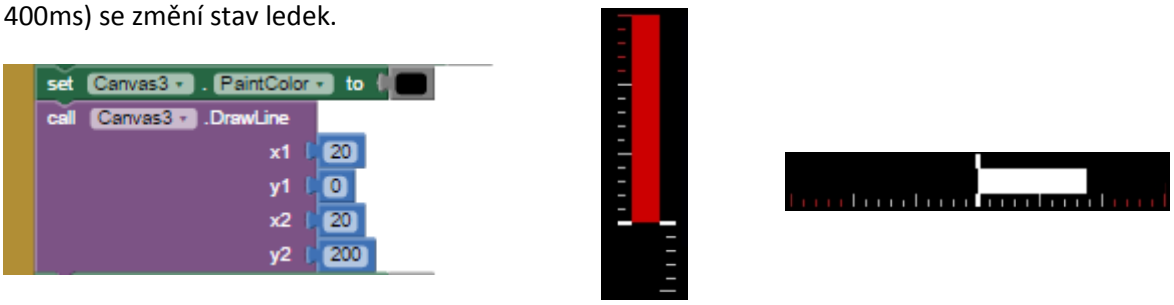
V další části události se jedná o změny jednotlivých grafických prvků a budou vysvětleny pouze několika ukázkami – konkrétně Ball1.MoveTo a Canvas3.Drawline.



Obrázek 39. Funkce Ball1.MoveTo a začátek if bloku posilat

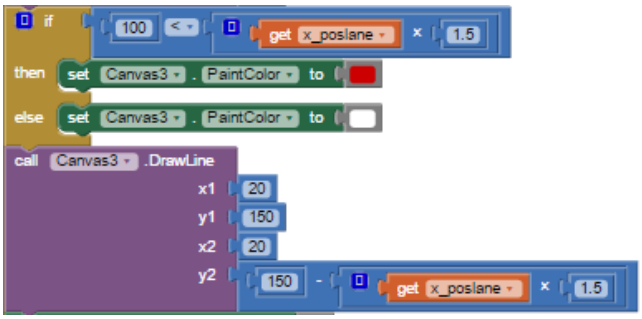
Funkce .MoveTo nad Ball1 mění aktuální pozici objektu Ball1 – její vstupy jsou souřadnice x a y s tím, že v levém dolním rohu je x a y rovno 0. Canvas1, na kterém je Ball1 umístěn má velikost 300x200 a souřadnice Ball1 jsou na něj přepočítány – „nulová“ pozice je ve středu Canvase1.

If blok, který je vidět dole na obr. 38, určuje, zda se údaje přes bluetooth budou nebo nebudou odesílat. Proměnná posílat je ovlivněna událostí Canvas1.TouchDown. Proměnná counter načítá počet odeslání – to je následně využíváno při blikání, kdy při každém čtvrtém odeslání (každých 400ms) se změní stav ledek.



Obrázek 40. Canvas3.DrawLine a ukázky použití

Jedna z velmi používaných grafických funkcí je pro vykreslování linií s určitou šířkou a délkou. Ukázka na obr. 39 je začernění před aplikací vlastního ukazatele. Změna barvy probíhá nastavením vlastnosti .paintColor na příslušnou barvu. Při použití tedy stačí nastavit barvu např. bílou nebo červenou a následně nastavit souřadnice v závislosti na momentálním natočení telefonu (použití proměnných x_poslane a y_poslane), viz obr. 39.



Obrázek 41. Canvas3.Drawline druhá ukázka

Dále jsou v události bloky ovládající blikání a přehrávání zvuků a blok pro tempomat a samozřejmě funkce odesílající data.

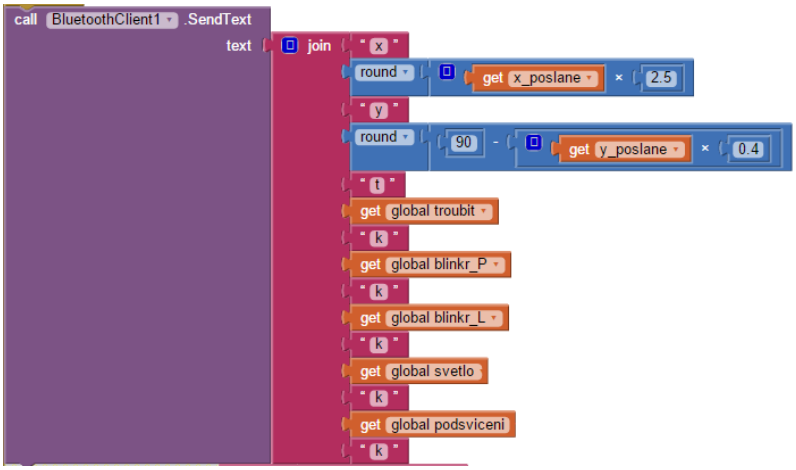
6.4.2.1. Funkce Bluetoothclient1.SendText

Funkce Bluetoothclient1.SendText odešle data ve formátu textového řetězce. Podstatné je tedy jaký formát textového řetězce zvolit a jak v něm jednotlivá data „kódovat“, vzhledem k tomu, že v Arduínu je pro preparování hodnot z přijatého řetězce využívána funkce ParseInt(), je možné

odesílat hodnoty Integer oddělené vždy nějakou non Integer hodnotou, v našem případě se jedná například o „x“,“y“,“t“,“k“.

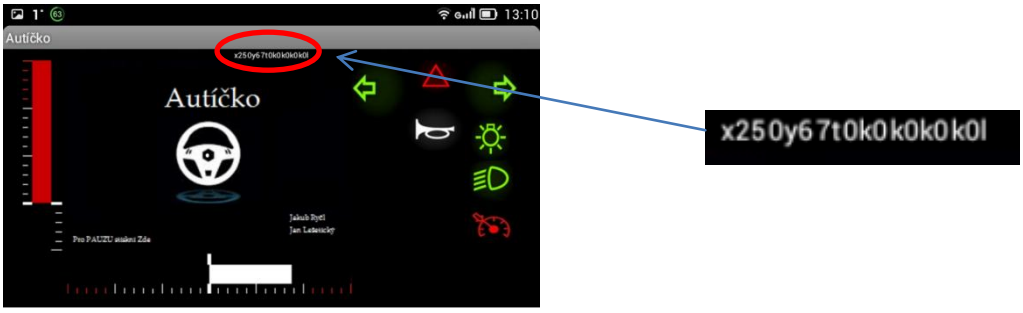
Posílané hodnoty (mezi jednotlivými oddělovači) postupně jsou:

- Otáčky motoru na hnací nápravě (rozsah -100 – 250)
- Pozice serva (rozsah 50 – 130, střed je 90 a na každou stranu +- 40)
- Klakson (spuštěn 1, vypnut 0)
- Pravý blinkr (spuštěn 1, vypnut 0)
- Levý blinkr (spuštěn 1, vypnut 0)
- Světlo (spuštěno 1, vypnuto 0)
- Podsvícení (spuštěno 1, vypnuto 0)

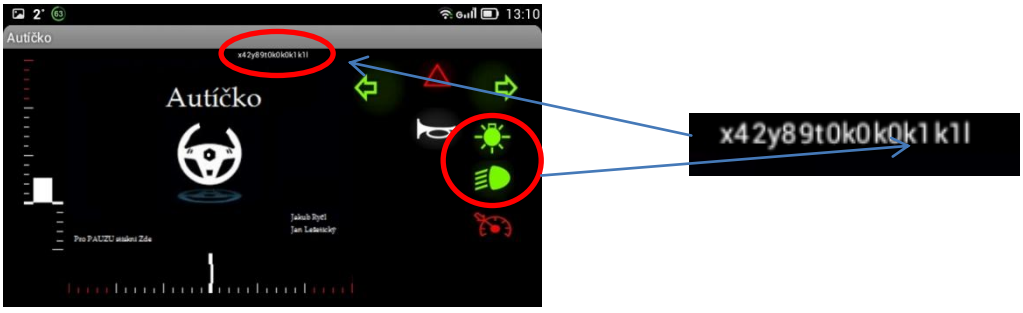


Obrázek 42 Funkce Bluetoothclient1.SendText

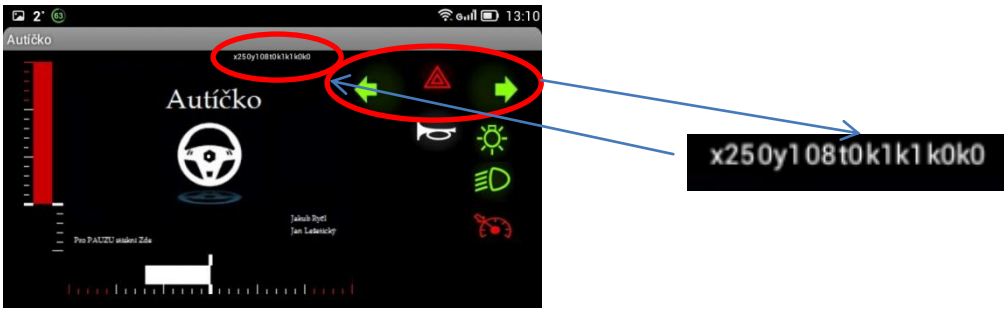
6.4.2.2. *Jednotlivé ukázky odesílaných dat (vygenerováno ze zkušební verze)*



Obrázek 43 Ukázka odesílaných dat 1 (maximální rychlost, zatočení vpravo)



Obrázek 44 Ukázka odesílaných dat 2 (zapnuté podsvícení a světla)



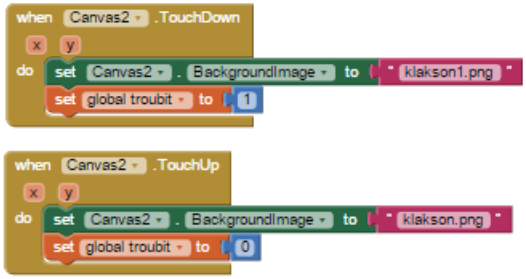
Obrázek 45 Ukázka odesílaných dat 3 (zapnutá výstražná světla)

6.5.Ovládání nástavbových funkcí

Jedná se o klakson, blinkry, výstražný trojúhelník, podsvícení, světla a tempomat.

6.5.1. Klakson

Klakson je spuštěn pouze při aktivním stisku, na jeho ovládání jsou tedy nutné dvě Události. Jedna při přiložení prstu (TouchDown)a druhá při jeho odložení (TouchUp).

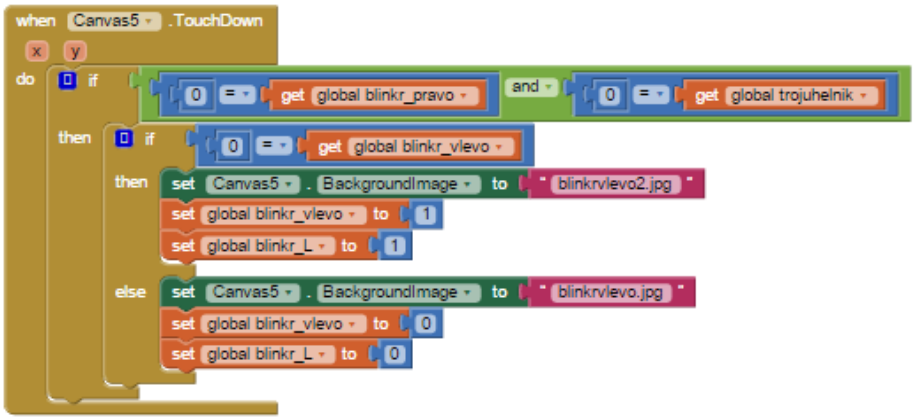


Obrázek 46. Ovládání Klaksonu

Uvnitř událostí se pouze mění zobrazovaný obrázek a proměnná troubit se mění z nuly na jedna a opačně.

6.5.2. Blinkr vpravo a vlevo

U blinkru se vyhodnocuje pouze stisk a na základě momentálního stavu se stav otočí (z nuly na jedna a opačně). Akce při stisknutí nastane, pouze pokud není zapnutý druhý blinkr nebo výstražná světla. Na obrázku 46. je možné vidět událost pro blinkr vlevo, pro blinkr vpravo je obdobná (změna na Canvas6).



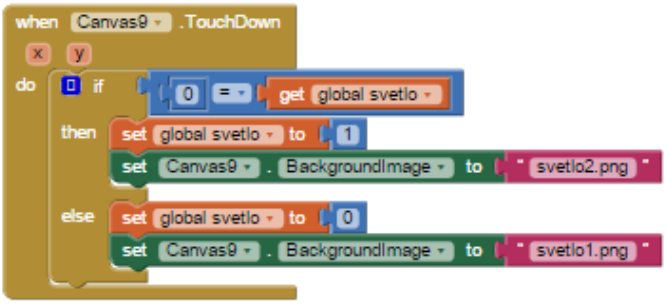
Obrázek 47. Blinkr vlevo

6.5.3. Výstražná světla

Výstražná světla fungují obdobně jako blinkr pouze na stisk (TouchDown). Při zapnutých světlech se nastavuje blinkr vpravo i vlevo na hodnotu jedna při vypnutých na 0.

6.5.4. Přední světla a podsvícení

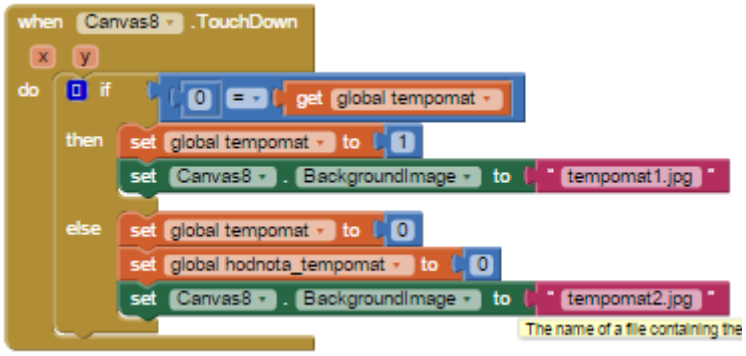
Zcela obdobně funguje i zapnutí předních světel a podsvícení na obr. 47 je pak ukázka pro přední světla.



Obrázek 48. Ovládání předních světel

6.5.5. Tempomat

Pro zapnutí tempomatu je funkčnost bloku také stejná, rozdíl je v bloku, který je umístěn uvnitř Události Clock1.Timer (na obr. 49). Pro funkčnost je nutné zavést dvě proměnné, první global tempomat rozhoduje o tom, zde je tempomat zapnutý nebo není, druhá global hodnota_tempomat do sebe ukládá hodnotu x_poslane při stisku Canvase8 a do opětovného stisku naopak přepisuje hodnotu x_poslane původně uloženou hodnotou v hodnota_tempomat.



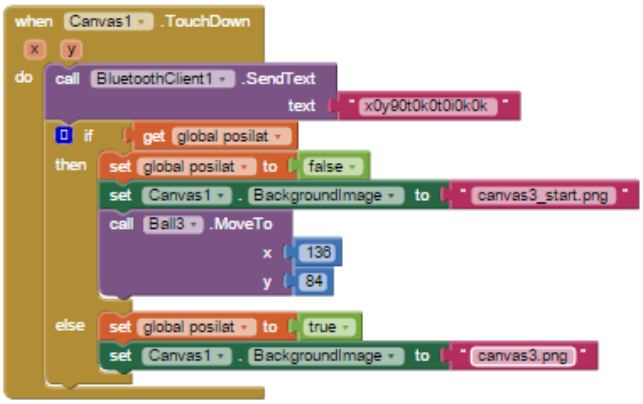
Obrázek 49. Tempomat

Obrázek 50. Blok pro tempomat v Clock1.Timer

6.5.6. Hlavní ovládací funkce

Ovládání samo o sobě začíná až po stisku ústřední obrazovky, kterou reprezentuje Canvas1. Při jeho stisku se vždy nejprve odešle nulový „inicializační“ text s hodnotou „x0y90t0k0t0i0k0k“ tedy

zastavení veškeré činnosti a následně se nastaví proměnná global posilat na true nebo false a změní se ústřední obrázek. Událost je možné vidět níže na obr. 50.



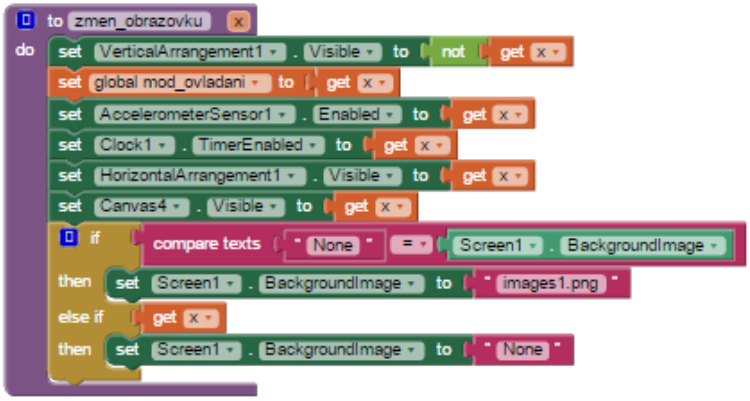
Obrázek 51. Canvas1.TouchDown

6.6.Interní funkce

Mezi takzvané interní funkce patří funkce, které souvisí s programovým zázemím aplikace a navenek se projevují jako dialogy, inicializace, atd.

6.6.1. Procedura zmen_obrazovku

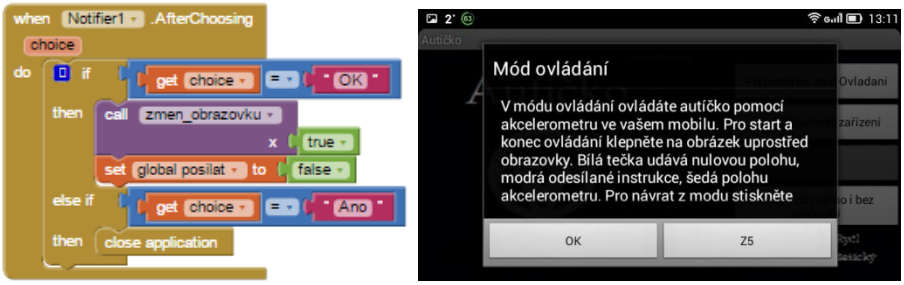
Procedura se používá ke snadnému měnění obrazovky z úvodní obrazovky na mód ovládání a nazpátek. Jako vstupní hodnota „x“ je do procedury převedena hodnota boolean (true/false).



Obrázek 52. Procedura zmen_obrazovku

6.6.2. Událost Notifier1.AfterChosing

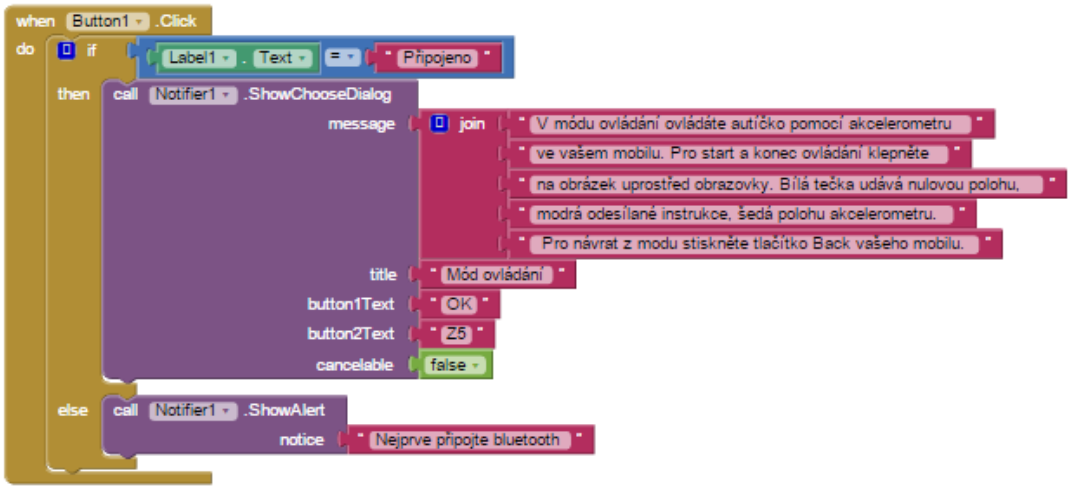
Událost je iniciována po vybrání odpovědi v Notifieru1.



Obrázek 53. Notifier1.AfterChosing

6.6.3. Událost Button1.Click

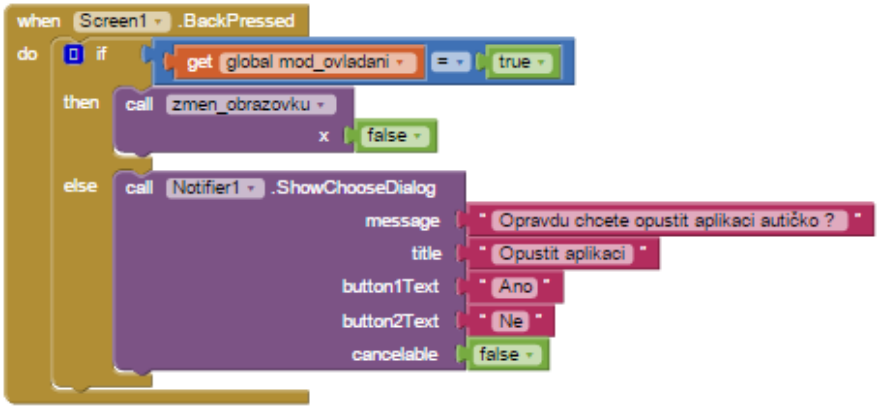
V případě, že je připojeno Bluetooth umožní přes informativní dialog přejít do Módu ovládání, v opačném vyzve k připojení BT.



Obrázek 54. Button1.Click

6.6.4. Událost Screen1.BackPressed

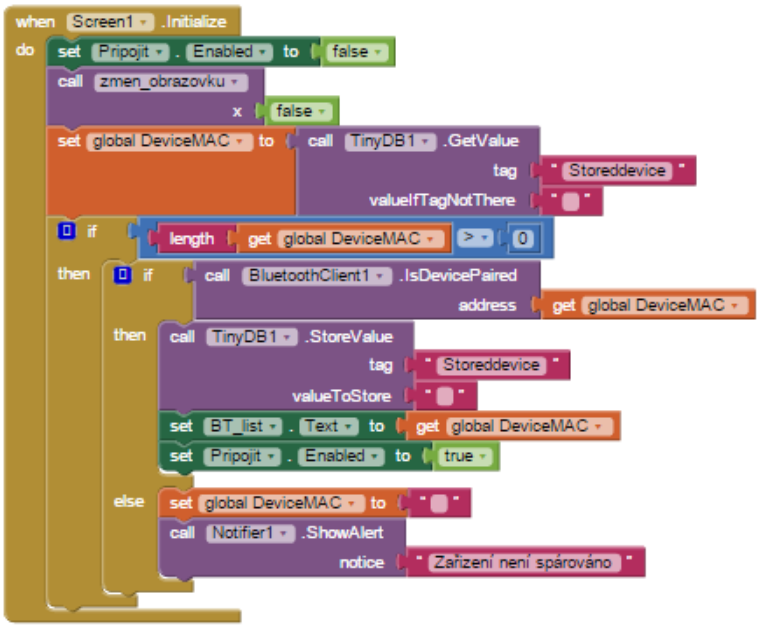
V případě stisku tlačítka Back (integrované tlačítko v každém mobilním telefonu)a v případě, že se uživatel nachází na úvodní obrazovce (mod_ovladani = false) a stiskne tlačítko je vyvolán Notifier1 pak při případné kladné odpovědi se dostáváme zpátky k události Notifier1.AfterChosing, kde je vyvolána akce *closeaplication.*



Obrázek 55 Událost Screen1.BackPressed

6.6.5. Událost Screen1.Initialize

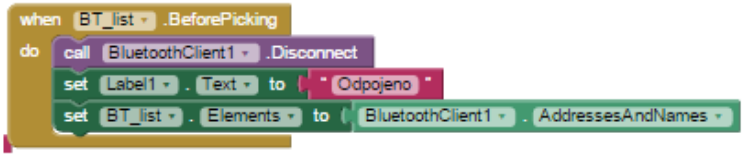
Inicializace obrazovky probíhá při zapnutí aplikace. Zde se také objevuje inicializace Bluetooth a TinyDB1 – což je databáze propojená s pamětí telefonu.



Obrázek 56. Inicializace

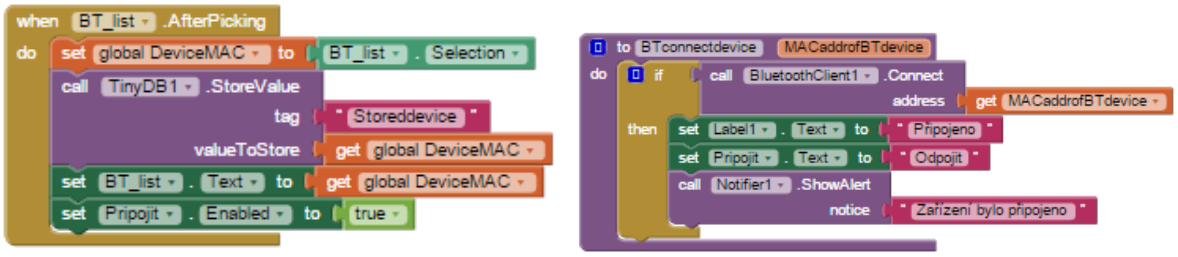
6.6.6. Programový postup připojení BT

Nejprve je nutné vybrat zařízení, které je nutné připojit, při stisku BT_listu (ale před tím než se cokoliv vybere) nastává událost BT_list.BeforePicking, kde se pro jistotu zařízení odpojí a do BT_listu se načtou adresy a jména z BluetoothClientu.



Obrázek 57. BT_list.BeforePicking

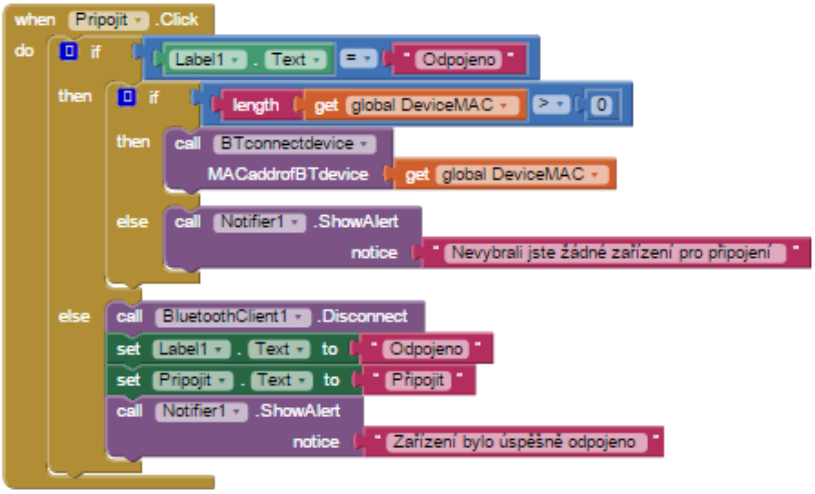
Následuje Událost AfterPicking, kde se do proměnné DeviceMAC uloží MAC adresa vybraného zařízení a ta je následně uložena do TinyDB1 pro další využití. Nyní je vybráno BT zařízení.



Obrázek 58. BT_list-AfterPicking a procedura BTconnectdevice

Dále je vhodné objasnit Proceduru *BTConnectDevice,* která připojuje BT zařízení. Zde je zajímavé především to, že v podmínce je samotná akce BTclient1.Connect1.Connect a připojí se na vybranou adresu. Tato procedura je využívána v Události Pripojit.Click.

Poslední nezbytnou součástí je Událost Pripojit.Click (obr 58.), kde se za předpokladu, že není zatím nic připojeno, zavolá procedura BTconnectDevice a naváže se spojení pomocí BT. V případě, že již spojení navázáno se naopak odpojí. V případě, že pro připojení není vybráno žádné zařízení, je uživatel upozorněn.



Obrázek 59. Pripojit.Click

7. Závěr

V semestrálním projektu byl vypracován a zdokumentován kompletní návrh ovládání autíčka přes bluetooth z mobilního telefonu. V průběhu vytváření práce bylo nutné skloubit mnoho různých odvětví (jak programování, elektroniku tak v neposlední řadě mechaniku), aby bylo docíleno výsledného funkčního modelu autíčka.

Citovaná literatura

[1] Arduino UNO R3 - CLON 100% + USB kabel. *Santy.cz* [online]. [cit. 2015-01-12]. Dostupné z: http://www.santy.cz/arduino-c2/arduino-uno-r3-nano-shield-atmel-328p-i47/

[2] HC Serial Bluetooth Products User Instructional Manual. In: *Rcscomponents* [online]. [cit. 2015-01-12]. Dostupné z: http://www.rcscomponents.kiev.ua/datasheets/hc_hc-05-user-instructions-bluetooth.pdf

[3] Ultrazvukový senzor vzdálenosti HC-SR04. *Santy.cz.* [online] [Citace: 2015-01-12.] http://www.santy.cz/moduly-c22/arduino-hc-sr04-modul-shield-mega-nano-vzdalenost-mereni-i23/

[4] Motor Driver Shield - 4 kanály (L293D). *Santy.cz* [online]. [cit. 2015-01-12]. Dostupné z:http://www.santy.cz/shieldy-pro-arduino-c23/motordrivershield-4ch-i195/

[5] Modul bzučáku, piezo siréna, kompatibilní s Arduino. *Easyduino.cz* [online]. [cit. 2015-01-12]. Dostupné z: http://www.easyduino.cz/Modul-bzucaku-piezo-sirena-kompatibilni-s-Arduino-d177.htm?tab=description

[6]Lekce-9-měříme-vzdálenost-s-hc-sr04. *Arduino8.webnode.cz* [online]. [cit. 2015-01-12]. Dostupné z: http://arduino8.webnode.cz/news/lekce-9-merime-vzdalenost-s-hc-sr04/

Seznam obrázků