MOBILNA APLIKACIJA ZA UPRAVLJANJE LED SVJETLIMA PLOČE ARDUINO UNO

Rešetar, Karlo

Undergraduate thesis / Završni rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Split / Sveučilište u Splitu**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:228:275433

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-05-18**

Repository / Repozitorij:

Repository of University Department of Professional Studies

zir.nsk.hr

UNIVERSITY OF SPLIT

dabar
DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U SPLITU

SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

KARLO REŠETAR

Z A V R Š N I R A D

MOBILNA APLIKACIJA ZA UPRAVLJANJE LED SVIJETLIMA PLOČE ARDUINO UNO

Split, rujan 2023.

SVEUČILIŠTE U SPLITU
SVEUČILIŠNI ODJEL ZA STRUČNE STUDIJE

Preddiplomski stručni studij Informacijska tehnologija

Predmet: Arhitektura i organizacija digitalnih računala

Z A V R Š N I R A D

Kandidat: Karlo Rešetar

Naslov rada: Mobilna aplikacija za upravljanje LED svjetlima ploče Arduino Uno

Mentor: dr. sc. Ivan Kedžo, prof. struč. stud.

Split, rujan 2023.

SADRŽAJ

Sažetak **1**

Summary **1**

1.Uvod **2**

2.Korištene Tehnologije **3**

2.1.Arduino IDE 3

2.2.Android Studio IDE 5

2.3.Kotlin 7

3.Mikrokontroler Arduino Uno **8**

3.1.Arduino Uno Ploča 8

3.1.1.PWM Pinovi 12

3.2.Bluetooth Modul HC-05 13

3.3.Programiranje Arduina 18

3.3.1.Uparivanje pametnog telefona s Bluetooth modulom 18

3.3.2.Arduino kôd 22

4.**Android mobilna aplikacija** **26**

4.1.Omogućavanje programerskih opcija 26

4.2.Izrada Android mobilne aplikacije 30

4.3.Definicija korisničkog sučelja za prikaz uparenih Bluetooth uređaja 31

4.4.Definicija korisničkog sučelja za upravljanje LED-icom 33

4.5.Funkcionalnost kôda android mobilne aplikacije 36

4.5.1.Spajanje android mobilne aplikacije bluetooth vezom 36

4.5.2.Kontroliranje LED diode android mobilnom aplikacijom 40

5.Zaključak **46**

Literatura **47**

Sažetak

U završnom radu napravljena je Android mobilna aplikacija korištenjem programskog jezika Kotlin za upravljanje Arduino Uno ploče sa svjetlećim diodama (engl. *light-emitting diode*). Aplikacija radi tako da uspostavlja kontakt između pametnog telefona i Arduino Uno ploče pomoću Bluetooth SPP (Serial Port Protocol) modula HC-05 koji je dizajniran za transparentno postavljanje bežične serijske veze, te omogućuje upravljanje svjetleće diode. Aplikacija na više načina omogućuje upravljanje svjetlećom diodom: paljenje i gašenje svjetleće diode, treptanje svjetleće diode te kontrola jačine svjetline (engl. *brightness control*) pomoću klizača (engl. *slider*). Za korisničko sučelje (engl. *user interface*) korišten je XML (EXtensible Markup Language) izgled ograničenja (engl. *constraint layout*) koji omogućuje pozicioniranje i veličinu malih aplikacija (engl. *widget*) na fleksibilan način. Također korištena je platforma GitHub za razvoj softvera i kontrolu verzija.

Ključne riječi: Android, Arduino Uno, Bluetooth, Kotlin, mobilna aplikacija

Summary

Mobile application for controlling the LED lights of the Arduino Uno board

In this final work, an Android mobile application was created using Kotlin programming language to control the light-emitting diode of the Arduino Uno board. The application works in a way by establishing a connection between the smartphone and the Arduino Uno board using the Bluetooth SPP (Serial Port Protocol) module HC-05, which is designed for transparent wireless serial communication, and enables control of the light emitting diode. The application provides multiple ways to manage the LED, such as turning it on and off, blinking LED, and brightness control using the slider. For the user interface XML (EXtensible Markup Language) constraint layout was used, which allows positioning and sizing of widgets in a flexible way. Also, GitHub is used for software development and version control.

Keywords: Android, Arduino Uno, Bluetooth, Kotlin, mobile application

1. Uvod

Zbog svakodnevnog korištenja pametnih telefona te zbog velikog porasta IoT (Internet of Things) [1] u svakodnevnome životu može se naići na razne pametne uređaje koji omogućavaju prikaz statusnih poruka (npr. kontrola temperature hladnjaka) čiji je princip rada baziran na postojanju internetske veze među uređajima. Stoga je u zadnjih nekoliko godina Bluetooth postao poznato ime u zajednici IoT kada je u pitanju lokalna kratkotrajna komunikacija između uređaja, pružajući brze, pouzdane i besprijekorne veze bez da se oslanjaju na internet. Cilj ovog završnog rada je prikazati kontroliranje svjetleće diode Arduino Uno ploče, od strane pametnog telefona, uz pomoć Bluetooth veze koja ih povezuje.

Završni rad se sastoji od pet poglavlja. U drugom poglavlju su opisane tehnologije koje su korištene u završnom radu. Treće poglavlje opisuje spajanje diode i Bluetooth HC-05 modula na Arduino Uno ploči te arhitekturu mikrokontrolera. U četvrtom poglavlju je prikazana konfiguracija postavki pametnog telefona koja je potrebna za izradu mobilne aplikacije, uključujući njen izgled, funkcionalnost i instalaciju. Također, opisano je na koji način mobilna aplikacija radi, te kontrolira svjetleću diodu na više načina. Na samom kraju dano je zaključno mišljenje o završnom radu.

2. Korištene Tehnologije

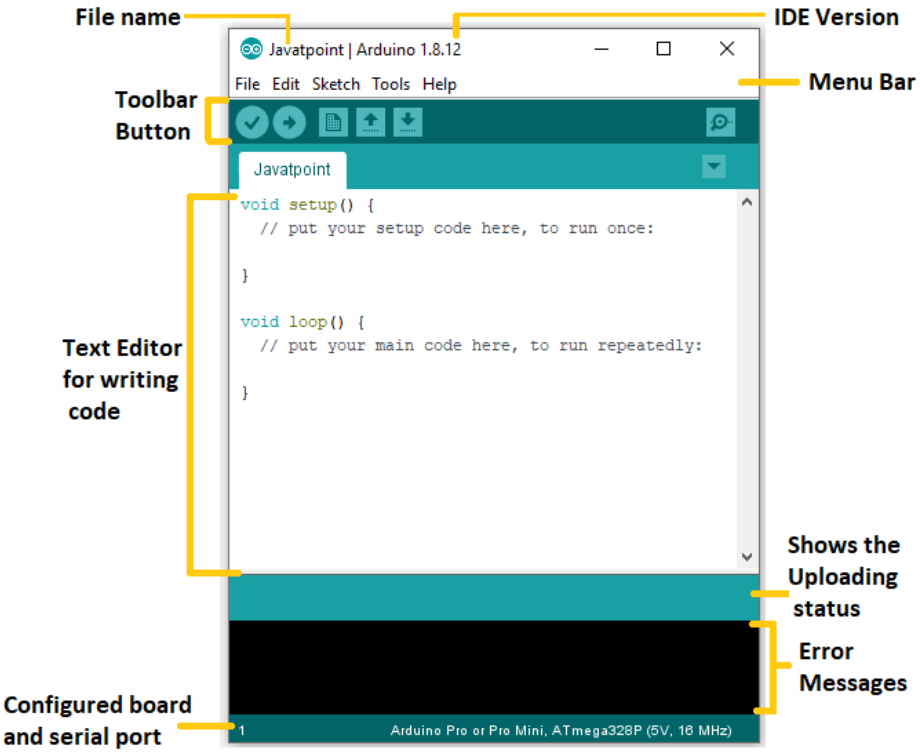
Mobilna aplikacija za upravljanje LED svjetlima ploče Arduino Uno zahtijevala je određene tehnologije. Prva predstavljena tehnologija u ovom završnom radu je Arduino IDE Integrirano razvojno okruženje (engl. *integrated development environment*) gdje je napisani kôd u programskom jeziku C služio kako bi se omogućio rad sa svjetlosnom diodom. Osim Arduino IDE razvojnog okruženja koristio se i Android Studio IDE, gdje je pisan kôd korištenjem programskog jezika Kotlin.

2.1. Arduino IDE

Arduino IDE je višeplatformska aplikacija koja je napisana u programskom jeziku Java. Uključuje uređivač kôda sa značajkama kao što su rezanje i lijepljenje teksta, pretraživanje i zamjena teksta, automatsko uvlačenje, podudaranje vitičastih zagrada i isticanje sintakse, te pruža jednostavne mehanizme jednim klikom za prevođenje i učitavanje programa na Arduino ploču.

Arduino IDE podržava jezike C i C++ koristeći posebna pravila strukturiranja koda. Isporučuje softversku biblioteku iz projekta Wiring, koja pruža mnoge uobičajene ulazne i izlazne procedure. Kod koji je napisao korisnik zahtijeva samo dvije osnovne funkcije, za pokretanje skice i petlje glavnog programa, koje se izvršavaju i povezuju s programskim zaglavljem `main()` u izvršni ciklički program. Također Arduino IDE koristi program avrdude koji omogućava pretvaranje izvršnog koda u tekstualnu datoteku u heksadecimalnom obliku koji se učitava u Arduino ploču od strane ugrađenog programa za učitavanje ploče. Između ostaloga važno je opisati skicu (engl. *Sketch*) u Arduino IDE-u, skice se spremaju na razvojno računalo kao tekstualne datoteke s nastavkom .ino. Također za minimalni Arduino C/C++ program potrebne su samo dvije funkcije. Prva funkcija je `setup()` koja se poziva jednom kada se skica pokrene, koristi se za inicijalizaciju varijabli, načina ulaza i izlaza pinova, a druga funkcija je `loop()` koja se izvršava više puta u glavnom

programu nakon što funkcija `setup()` završi te također upravlja pločom dok se ploča ne isključi ili resetira. Razvojno okruženje Arduino IDE je prikazano na slici 1.



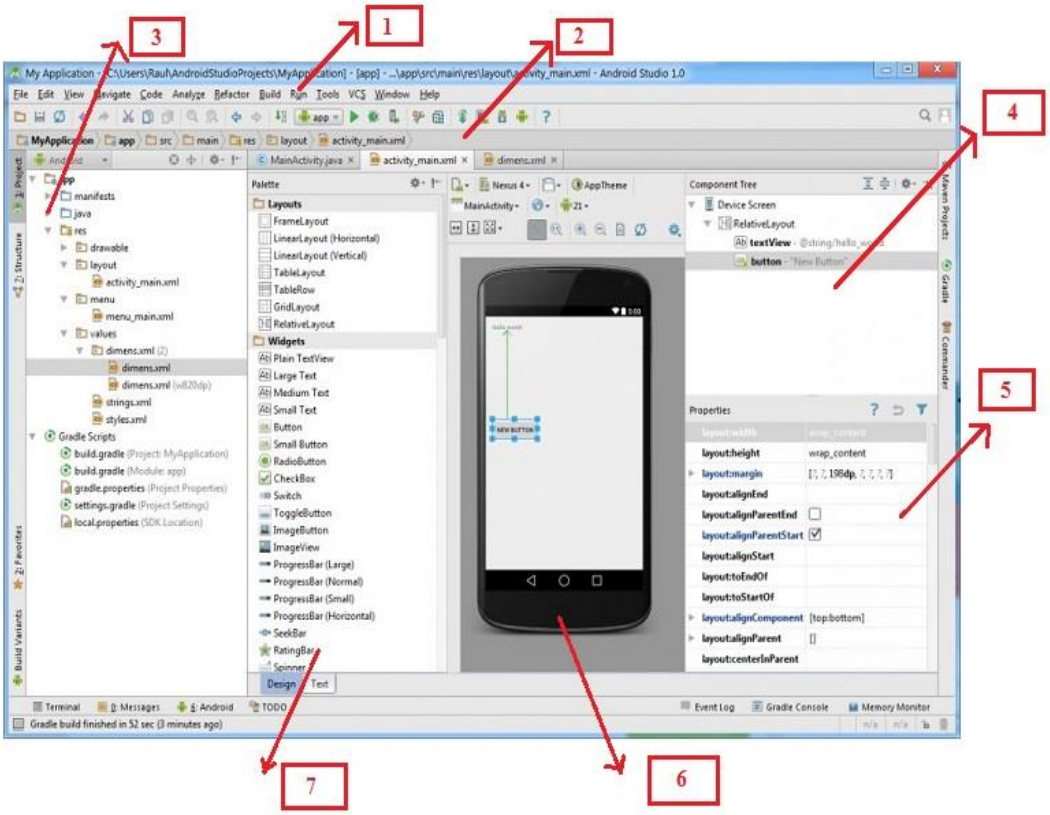
Slika 1: Razvojno okruženje Arduino IDE [2]

2.2. Android Studio IDE

Android Studio je službeni IDE za Googleov operativni sustav Android, napravljen na JetBrainsovom IntelliJ IDEA softveru i dizajniran posebno za razvoj Android aplikacija. Dostupan je za preuzimanje na Windowsu, macOS-u i Linuxu operativnim sustavima. Android Studio je najavljen 16.svibnja 2013. godine na Google konferenciji gdje je bio u ranoj fazi (engl. *early access*), potom je u lipnju 2014. godine objavljena beta faza verzije 0.8 te je prva stabilna verzija objavljena u prosincu 2014. godine. Krajem 2015. godine Google je ukinuo podršku za Eclipse ADT (Android Development Tools) zbog čega je postao službeni IDE za razvoj Android aplikacija.

Značajke Android Studija su da je fleksibilan sustav izrade temeljen na Gradleu, ima brzo oponašanje pametnog telefona (engl. *emulator*), okruženje je jedinstveno u kojem se mogu razvijati aplikacije za sve Android uređaje, također postoji uživo uređivanje i ažuriranje komponenti na fizičkim uređajima i u oponašateljima pametnih telefona, nudi predloške kôda i GitHub integraciju koja pomaže kod izrade značajki mobilne aplikacije i za uvoz primjerka kôda, podržava C++ i NDK te ima ugrađenu podršku Googleove platforme koja olakšava integraciju.

Na slici 2 je opisano razvojno okruženje glavnog dijela korisničkog sučelja Android Studija.



Slika 2: Razvojno okruženje Android Studija[4]

Crvene oznake prikazuju sljedeće:

1) Alatna traka - Skup mnogih alata kao što je rezanje, kopiranje, lijepljenje, pokretanje otklanjanja pogrešaka (engl. *run debug*) i mnogo drugih mogućnosti.

2) Navigacijska traka - Pomaže u navigaciji kroz nedavno otvorene datoteke projekta.

3) Struktura projekta - Struktura mapa stvorenog projekta.

4) Stablo komponenti - Prikazuje komponentu koja se koristi u aktivnosti projekta u obliku strukture stabla.

5) Prozor svojstava - Prikazuje svojstva odabrane stavke na ekranu.

6) Uređivač izgleda - Prikazuje grafički izgled kako će aplikacija izgledati.

7) Prozor paleta - Prikazuje komponente, izglede i *Widgete* dostupne u Android Studiju.

2.3. Kotlin

Kotlin je programski jezik otvorenog kôda (engl. *open-source*) koji podržava objektno, ali i funkcionalno programiranje. Kotlin nudi sličnu sintaksu kao i koncepte iz drugih jezika, primjerice C#, Java, Scala i mnogi drugi. Programski jezik Kotlin ne teži biti jedinstven štoviše, uzima inspiraciju od drugih razvoja jezika proteklog desetljeća. Postoji u varijantama koje ciljaju na JVM (*Java virtual machine*) (Kotlin/JVM), JavaScript (Kotlin/JavaScript) i izvorni kôd (Kotlin/Native). Kotlinom upravlja Zaklada Kotlin, grupa koju su osnovali JetBrains i Google, a zadatak im je unaprijediti i nastaviti razvijati jezik. Također Google službeno podržava Kotlin za Android razvoj, što znači da su dokumentacija i alati za Android dizajnirani imajući na umu programski jezik Kotlin. Određeni Android API-ji (*Application programming interface*), kao što je Android KTX, specifični su za Kotlin, ali većina ih je napisana u programskom jeziku Java i mogu se pozivati iz Jave ili Kotlina. Kotlinova kompatibilnost s Javom ključna je za njegov rast. Što je omogućeno da se Java kôd može pozivati iz Kotlina i obrnuto. Popularnost Kotlina brzo raste te rezultira ljepšim iskustvom razvoja na Androidu, ali razvoj Androida nastavlja se imajući na umu i Kotlin i Javu. Na slici 3 prikazana je usporedba nekih od značajki programskog jezika Java i Kotlina.

Features	Kotlin	Java
Fully OPP (Object Oriented Programming)	Fully OPP	Not Pure OPP
Functional Reactive Programming	Yes	No
Lambda Expression	Yes	No
NULL Safety	Yes	No
Invariant Array	Yes	No
Smart Casts	Yes	No
Singleton Objects	Easily create Singleton Objects	Yes
Checked Exception	No	Yes

Slika 3: Usporedba značajki programskih jezika Kotlin/Java [7]

3. Mikrokontroler Arduino Uno

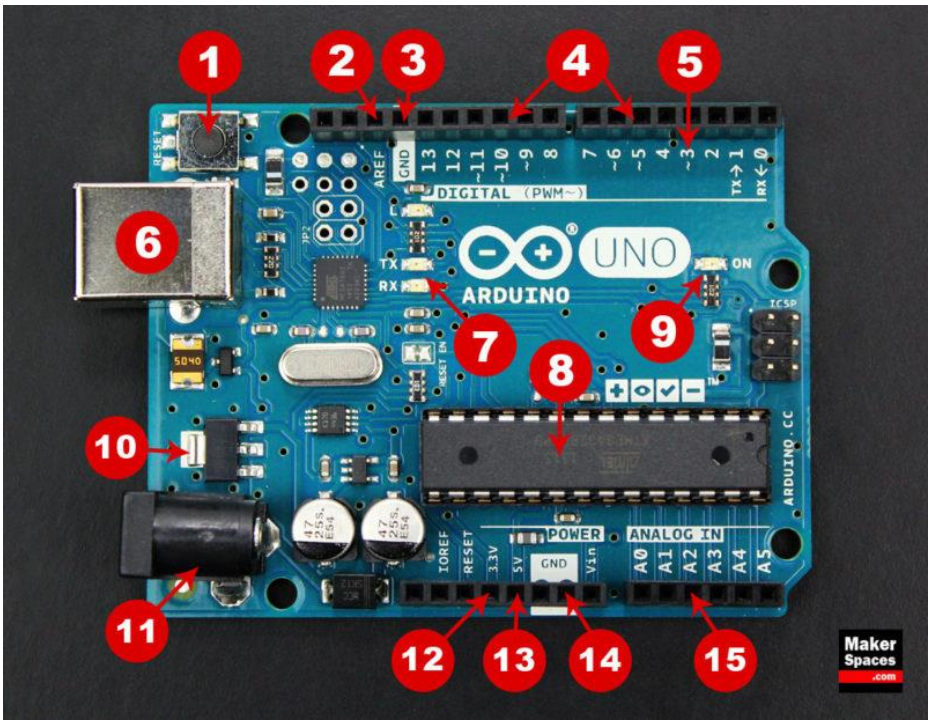
U ovom poglavlju je opisano spajanje Arduino Uno ploče s Bluetooth modulom i LED diodom uz pomoć Eksperimentalne pločice (engl. *breadboard*) te opisivanje rada sa samim mikrokontrolerom kako bi se kontrola svjetleće diode uspješno izvršila.

Komponente koje su bile potrebne za upravljanje LED svijetlima ploče Arduino UNO su:

- Arduino Uno Ploča
- USB kabel tipa B
- Spojne žice
- LED dioda
- Bluetooth Modul HC-05
- Eksperimentalna pločica

3.1. Arduino Uno Ploča

Arduino Uno ploča je mikrokontroler otvorenog kôda koji je temeljen na Microchip ATmega328P mikrokontroleru koju je razvio Arduino.cc i prvi put je objavljen 2010. godine. Ploča mikrokontrolera opremljena je setovima digitalnih i analognih ulazno/izlaznih pinova koji se mogu povezati s različitim pločama za proširenje (engl. *Shields*) i drugim sklopovima. Arduino Uno ploča ima sve skupa 14 digitalnih ulazno/izlaznih pinova od kojih su 6 pinova sposobni za korištenje kao PWM (Pulse Width Modulation) izlazi, također ploča ima 6 analognih ulazno/izlaznih pinova i može se programirati s Arduino IDE putem USB kabela tipa B. Na slici 4 prikazana je slika ploče Arduino Uno.



Slika 4: Arduino UNO ploča[8]

1) Botun za resetiranje - ponovno pokreće kôd koji je učitan u Arduino ploču.
2) AREF - analogna referenca (engl. *analog reference*) koristi se za postavljanje vanjskog referentnog napona.
3) Pin za uzemljenje (GND) - na Arduino ploči postoji nekoliko pinova za uzemljenje i svi rade istu funkciju.
4) Digitalni ulaz/izlaz - pinovi od 0 do 13 mogu se koristiti za digitalni ulaz ili izlaz.
5) PWM - pinovi (3, 5, 6, 9, 10, 11) koji su označeni s PWM (~) mogu simulirati analogni izlaz.
6) USB veza - koristi se za napajanje Arduina i učitavanja napisanog kôda u Arduino.
7) TX/RX - LED indikatori prijenosa i primanja podataka.
8) ATmega mikrokontroler - mjesto gdje se pohranjuju programi te sami mozak ploče.

9) LED indikator napajanja - LED indikator koji svijetli svaki put kada je ploča priključena na izvor napajanja.
10) Regulator napona - kontrolira količinu napona koja ulazi u ploču.
11) DC napajanje Jack - koristi se za napajanje Arduina.
12) Pin od 3,3 V - služi za napajanje 3,3 volta kod projekata.
13) Pin od 5V - služi za napajanje 5 volta kod projekata.
14) Pinovi za uzemljenje (GND)
15) Analogni pinovi - pinovi koji mogu očitati signal s analognog senzora i pretvoriti ga u digitalni signal.

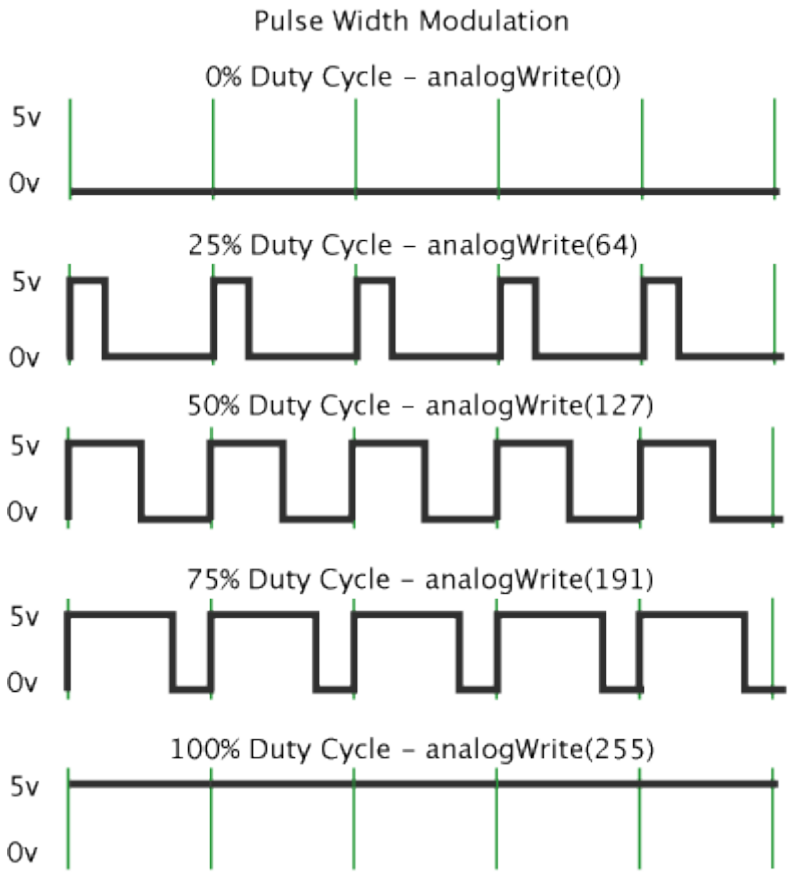
Tablica 1: Tehničke karakteristike Arduino UNO ploče.

MIKROKONTROLER	ATmega328P
RADNI NAPON	5 V
PREPORUČENI ULAZNI NAPON	7 - 12 V
LIMITIRANI ULAZNI NAPON	6 - 20 V
DIGITALNI ULAZNO/IZLAZNI (I/O) PINOVI	14 (od kojih 6 pinova omogućavaju PWM izlaz)
PWM DIGITALNI I/O PINOVI	6
ANALOGNI ULAZNI PINOVI	6
DC(ISTOSMJERNA) STRUJA PO I/O PINOVIMA	20 mA
DC STRUJA ZA 3.3 V PINOVE	50 mA
FLASH MEMORIJA	32 KB (Atmega328P) od čega 0.5 KB je korišteno od strane pokretačkog programa (engl. *boot loader*)
SRAM	2 KB (ATmega328P)
EEPROM	1 KB (ATmega328P)
BRZINA	16 MHz
DULJINA	68.6 mm
ŠIRINA	53.4 mm
TEŽINA	25 g

3.1.1. PWM Pinovi

U ovom završnom radu PWM (Pulse Width Modulation) pinovi su iskorišteni radi kontroliranja jačine svjetlosti LED diode. PWM radi tako da se digitalna kontrola koristi za stvaranje kvadratnog vala, signala koji se uključuju i isključuju. U sljedećem primjeru pokazano je kako PWM zaista funkcionira gdje je prikazan primjer za blijeđenje (engl. *Fading*) LED diode. Signal koji se uključuje i isključuje može simulirati napone između punog Vcc (minimalna snaga potrebna za učinkovit rad Arduina) i isključenog Vcc. Širina pulsa je trajanje „na vrijeme“ (engl. *„on time“).* Da bi se dobile različite analogne vrijednosti, mijenja se ili modulira širina tog impulsa. Ako se ponovi uzorak uključivanja-isključivanja dovoljno brzo s LED diodom rezultat je sličan kao da je signal stalan napon između 0 i Vcc koji kontrolira svjetlinu LED diode.

Na slici 5 zelene linije predstavljaju pravilno vremensko razdoblje. To trajanje ili razdoblje je suprotno od PWM frekvencije. S Arduino PWM frekvencijom na oko 500 Hz, zelene linije bi bile međusobno razmaknute 2 milisekunde. Funkcija `analogWrite()` služi za zapisivanje analognih vrijednosti na PWM pinu. Također `analogWrite()` je u rasponu 0-255, tako da `analogWrite(255)` zahtjeva 100% radni ciklus (uvijek uključen), dok je na primjer `analogWrite(127)` 50% radnog ciklusa.

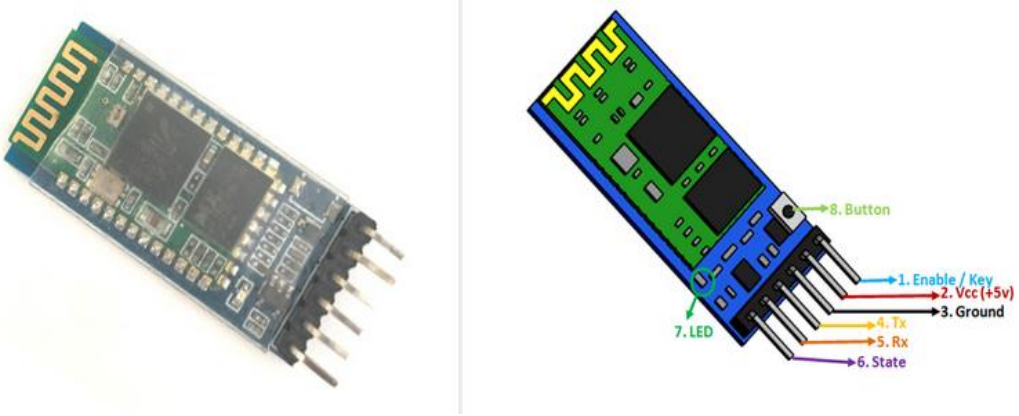


Slika 5: Modulacija širine impulsa[9]

3.2. Bluetooth Modul HC-05

HC-05 je popularan bluetooth modul koji u projektima omogućuje te može dodati dvosmjernu bežičnu funkcionalnost. Može se koristiti za komunikaciju između dva mikrokontrolera kao što je Arduino ili za komunikaciju s bilo kojim uređajem koji posjeduje bluetooth funkciju kao što je pametni telefon ili prijenosno računalo. Modul komunicira uz pomoć USART-a (Univerzalni sinkroni i asinkroni prijemnik-odašiljač, koji je vrsta uređaja za asinkronu ili sinkronu komunikaciju) pri brzini od 9600 prijenosa podataka (engl. *baud rate)*, stoga ga je lako spojiti s bilo kojim mikrokontrolerom koji podržava USART. HC-05 ima dva načina rada, jedan od njih je podatkovni način rada koji služi za slanje i primanje

podataka s drugih bluetooth uređaja, a drugi je AT način rada (engl. *Attention mode*) gdje se mogu promijeniti zadane postavke samoga uređaja. Modul HC-05 je vrlo jednostavno upariti s mikrokontrolerima radi protokola serijskog porta (SPP), gdje se šalje napajanje modulu +5V ili +3.3V i spoj između RX pina i TX pina na mikrokontroleru te TX pina na RX pin mikrokontrolera. Na slici 6 prikazan je bluetooth modul HC-05 te njegovi izvodi prikazani su izvodi (engl. *pinout*) modula.



Slika 6: HC-05 Bluetooth modul / izvodi modula[10]

Tablica 2: Konfiguracija izvoda HC-05 modula

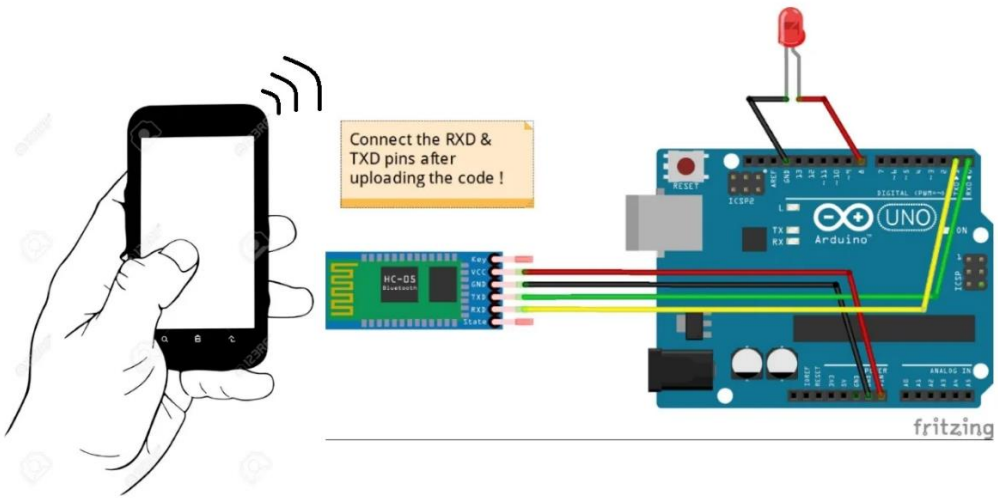
Broj PINA	**Ime PINA**	**Opis**
1	Omogućiti / Tipka	Ovaj pin se koristi za prebacivanje između podatkovnog načina koji je postavljen nisko (engl. *set low*) i AT naredbenog načina koji je postavljen visoko (engl. *set high*).
2	Vcc	Napaja modul.
3	Uzemljenje	Pin za uzemljenje.
4	TX odašiljač	Sve podatke koje primi putem bluetootha, ovaj pin će dati kao serijske podatke.
5	RX prijamnik	Svaki serijski podatak koji RX pin prima bit će emitiran putem bluetootha.
6	Stanje (engl. *State*)	Ovaj pin se može koristiti kao povratna informacija za provjeru je li Bluetooth radi ispravno.

7	LED	Označava status modula: • Ako modul treperi svako 2 sekunde: Modul je naredbenom načinu rada (engl. *command mode*) • Ako se treptanje ponavlja: Čeka se veza u podatkovnom načinu rada • Ako modul treperi dvaput u 1 sekundi: Veza je uspješna u podatkovnom načinu rada
8	Botun	Koristi se za upravljanje tipkom omogućiti (engl. *enable*) za prebacivanje između podatkovnog i naredbenog načina rada

Tehničke karakteristike HC-05 modula:

- Serijski Bluetooth modul za Arduino i druge mikrokontrolere
- Radni napon: 4V do 6V (obično +5V)
- Radna struja: 30mA
- Domet: <100m
- Radi sa serijskom komunikacijom te je TTL(tranzistor-tranzistorski logički sklop) kompatibilan
- Prati standardizirani protok IEEE 802.15.1
- Koristi frekventno skakanje širenja spektra (FHSS)
- Može raditi u Master, Slave ili Master/Slave načinu rada
- Jednostavno se povezuje s prijenosnim računalom ili pametnim telefonom s Bluetoothom
- Podržana brzina prijenosa: 9600, 19200, 38400, 115200, 230400, 460800

Na slici 7 prikazan je shematski prikaz Arduina UNO, HC-05 modula i svjetleće diode.



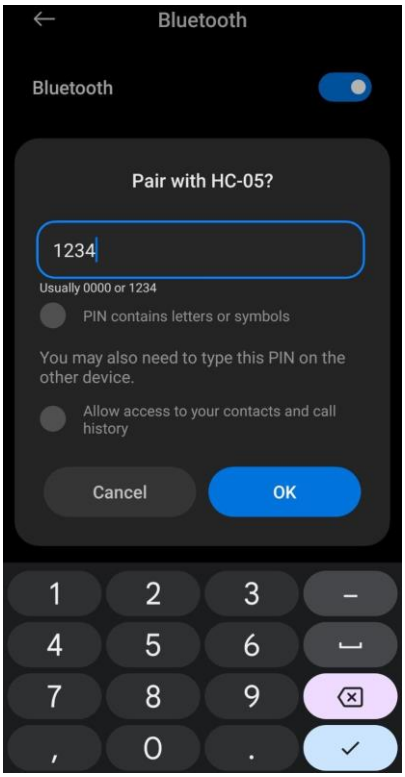
Slika 7: Shematski prikaz Arduino UNO, HC-05, svjetleća dioda[11]

3.3. Programiranje Arduina

Prije objašnjenja kôda Arduino aplikacije potrebno je objasniti kako spojiti pametni telefon s Bluetooth HC-05 modulom.

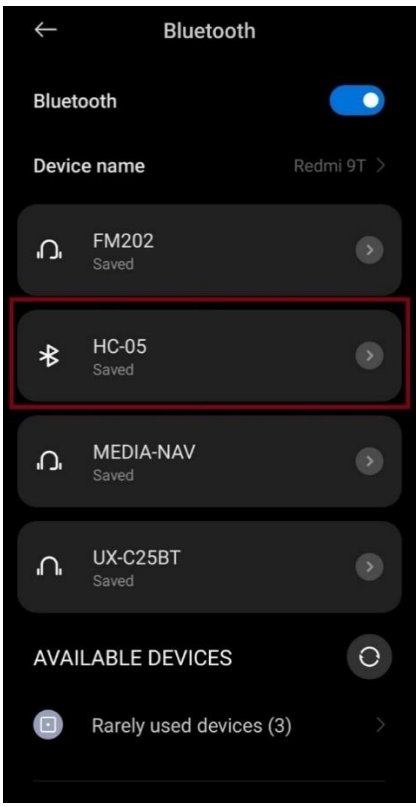
3.3.1. Uparivanje pametnog telefona s Bluetooth modulom

Prvi korak je spojiti Arduino Uno pomoću USB kabela na prijenosno računalo, zatim na mobilnom telefonu osposobiti Bluetooth. Nakon toga, ulaskom u prozor Bluetootha gdje su prikazani svi pronađeni Bluetooth uređaji i pritiskom na ime uređaja (u ovom slučaju HC-05) otvara se novi prozor gdje se upiše PIN. PIN je po zadanim postavkama HC-05 Bluetooth modula postavljen na 1234 te se nakon njegovog upisa klikom na tipku OK uspostavlja veza između pametnog telefona i HC-05 Bluetooth modula. Na slici 8 prikazan je postupak.



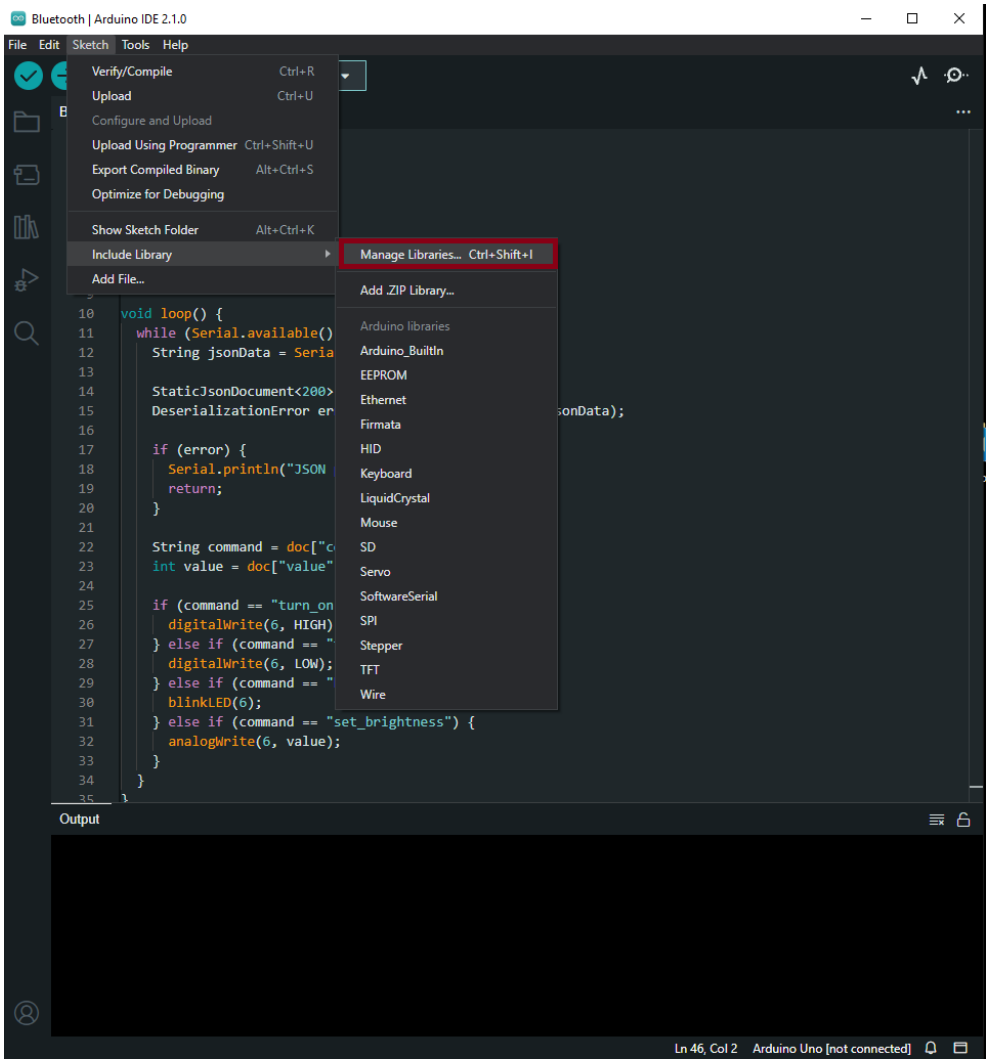
Slika 8: Postupak spajanja pametnog telefona s HC-05 Bluetooth modulom

Kada se svi koraci izvrše ispravno, pametni telefon je uspješno spojen s HC-05 Bluetooth modulom te je prikazan u listi spojenih uređaja. Na slici 9 prikazan je rezultat uspješno spojenog HC-05 Bluetooth modula s pametnim telefonom.



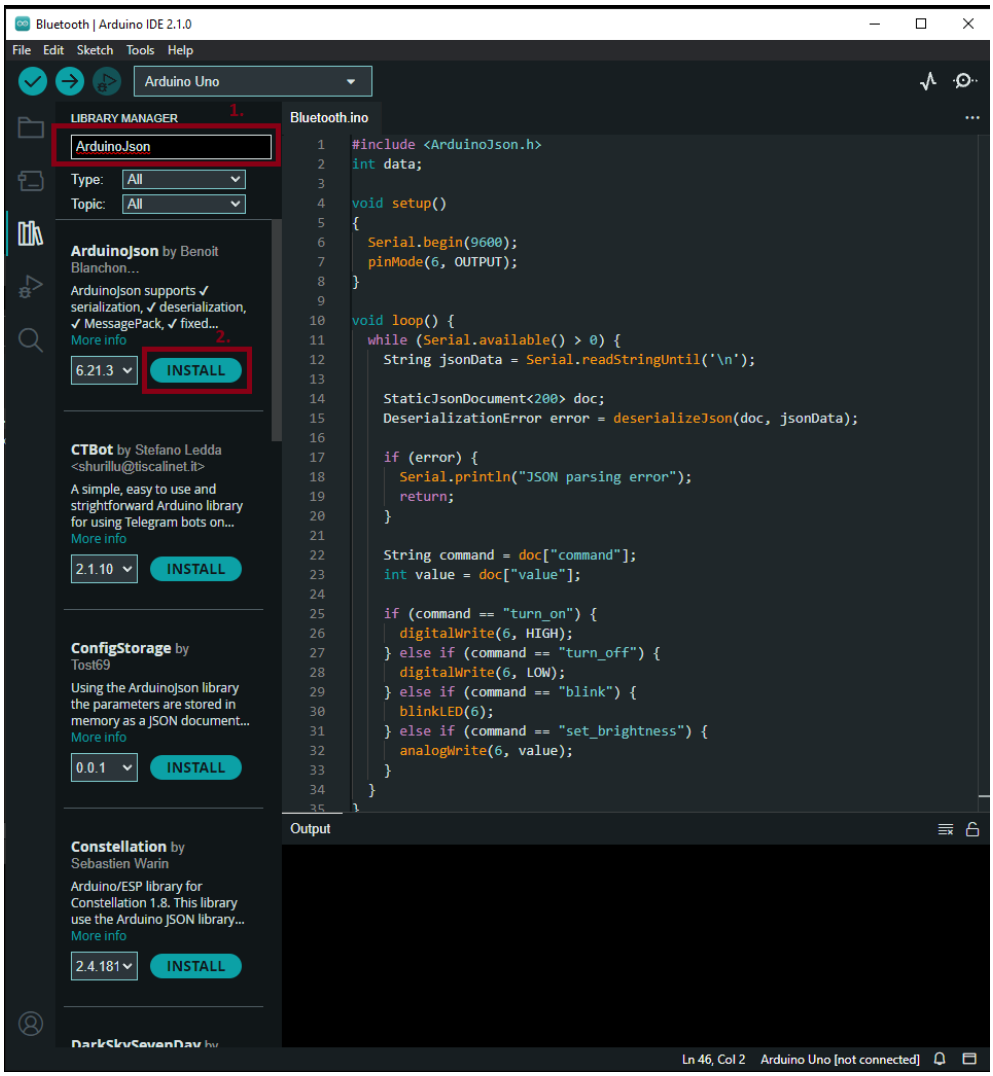
Slika 9: Uspješno spojen HC-05 Bluetooth modul s pametnim telefonom

Komunikacija Arduina i Android mobilne aplikacije vrši se pomoću JSON (JavaScript Object Notation) tekstualnog formata. JSON je otvoreni standardni format datoteke i format za razmjenu podataka koji koristi tekst čitljiv za pohranu i prijenos podatkovnih objekata. Potrebno je instalirati ArduinoJson[12] biblioteku unutar Arduino IDE-a. Prvi korak je prikazan na slici 10.



Slika 10: Prvi korak instalacije ArduinoJson biblioteke

Na alatnoj traci nalazi se opcija Skica (engl. *Sketch*) klikom na opciju sketch otvara se prozor s pojedinim opcijama, a jedna od njih je uključivanje biblioteke. Dolaskom kursora na opciju proširuje se izbor opcija. Opcija koja je potrebna za instalaciju je upravljanje bibliotekama (engl. *Manage Libraries*) te se klikom na nju otvara novi prozor s lijeve strane (Slika 11).



Slika 11: Prozor upravljanja bibliotekama

Sljedeći korak je u traku za pretraživanje (engl. *Search bar*) upisati: ArduinoJson te klikom na instaliraj (engl. *install*) uspješno je instalirana ArduinoJson biblioteka.

3.3.2. Arduino kôd

U ovom poglavlju bit će objašnjen Arduino kôd koji je napisan da bi mobilna aplikacija uspješno kontrolirala svjetleću LED diodu.

```
#include <ArduinoJson.h>
void setup()
{
  Serial.begin(9600);
  pinMode(6, OUTPUT);
}
```

Ispis 1: Potrebne funkcije za komunikaciju s modulom i LED diodom

Kao što je već objašnjeno u prethodnim poglavljima, `void setup()` je prva funkcija koja će biti pokrenuta samo jednom te pokreće napisani kôd unutar Arduina. U prvoj liniji kôda je deklarirana biblioteka `<ArduinoJson.h>` jer se komunikacija s mobilnom aplikacijom vrši preko JSON (JavaScript Object Notation) tekstualnog formata. Zatim je korištena funkcija `Serial.begin(9600)` za inicijalizaciju komunikacije pri brzini 9600 prijenosa podataka koja služi za serijsku komunikaciju s HC-05 Bluetooth modulom te omogućuje korištenje funkcije `Serial.read()` za primanje podataka od strane spojenog uređaja. Funkcija `pinMode(pinNumber, mode)` koja prima 6 kao prvi parametar i ukazuje na broj pina na Arduino ploči te OUTPUT kao drugi parametar koji omogućuje konfiguraciju digitalnog pina 6 kao izlaz (engl. *output*), što znači da omogućava slanje električnih HIGH (engl. *visok*i) / LOW (engl. *niski*) signala za kontroliranje LED diode. HIGH u ovom slučaju služi za uključivanje LED diode, dok LOW služi za isključivanje LED diode, također razlog zbog kojeg je korišten pin 6 je taj jer podržava PWM.

```
void loop() {
  while (Serial.available() > 0) {
    String jsonData = Serial.readStringUntil('\n');
    StaticJsonDocument<64> doc;
    DeserializationError error = deserializeJson(doc, jsonData);
```

Ispis 2: Arduino kôd za deserijalizaciju podataka

Funkcija `void loop()` služi za pisanje glavnog kôda programa koji sadržava logiku te se pokreće nakon što se funkcija `setup()` izvrši to jest nakon što se funkcija `setup()` jednom pokrene.

`While(Serial.available() > 0)` je petlja koja provjerava to jest dobiva pohranjene bajtove sa serijskog priključka koji su dostupni za čitanje. To su podaci koji su već pohranjeni i stigli u serijski međuspremnik (engl. *buffer*) (serijski međuspremnik u Arduinu sadrži 64 bajta). Sljedeća linija kôda gdje je deklarirana varijabla jsonData služi za pohranu podataka od strane mobilne aplikacije, te čita primljeni podatak u ovome slučaju niz znakova (engl. *String*) sve dok ne dođe do kraja stringa (oznaka za kraj stringa je '\n'). Linija `StaticJsonDocument<kapacitet>` je klasa koju omogućuje biblioteka ArduinoJson, omogućuje stvaranje objekta JSON dokumenta u memoriji koji se koristi za rukovanje i generiranje JSON podataka. Zatim <kapacitet> predstavlja broj bajtova koji su alocirani za JSON dokument te određuje koliko memorije JSON dokument može koristiti za pohranu. Deserijalizacija je proces pretvaranja podatkovne strukture ili objekta u niz bajtova za pohranu ili za prijenos preko uređaja. Na sljedećem ispisu 3 prikazan je nastavak Arduino kôda.

```
String command = doc["command"];
    int value = doc["value"];
    if (command == "turn_on") {
      digitalWrite(6, HIGH);
    } else if (command == "turn_off") {
      digitalWrite(6, LOW);
    } else if (command == "blink") {
      blinkLED(6);
    } else if (command == "set_brightness") {
      analogWrite(6, value);
    }
  }
}
```

Ispis 3: Arduino kôd za kontroliranje LED diode

Ovaj dio kôda omogućava različite funkcionalnosti kao što su: uključivanje, isključivanje, treptanje i kontrola jačine svjetlosti LED diode. Radi tako da `doc[„command“]` izvlači vrijednosti iz JSON dokumenta i sprema ih u varijablu command, koristi se za naredbe za uključivanje, isključivanje, treptanje i kontroliranje svjetlosti LED diode, dok varijabla value uzima prima vrijednost s mobilne aplikacije i šalje se u funkciju `analogWrite()` za zapisivanje analogne vrijednosti i mogućnosti osvjetljivanja LED diode.

```
void blinkLED(int pin)
{
  for (int i = 0; i < 3; i++)
  {
    digitalWrite(pin, HIGH);
    delay(500);
    digitalWrite(pin, LOW);
    delay(500);
  }
}
```

Ispis 4: Funkcija za treptanje LED diode

Definicija funkcije za treptanje LED diode `blinkLED()` radi tako da LED dioda zasvijetli te ima kašnjenje (engl. *delay*) od 0,5 sekundi to jest 500 milisekundi. U konačnici Arduino prima JSON tekstualnu datoteku i provjerava je li s mobilne aplikacije dostavljen jedan od Stringova: „turn_on“, „turn_off“, „blink“ ili „set_brightness“. Ako je, Arduino odrađuje svoje funkcije kao što je već spomenuto: uključivanje, isključivanje, treptanje ili kontroliranje svjetlosti LED diode. Također kada se Arduino kôd prilikom učitavanja (engl. *upload*) iz Arduino IDE-a šalje na Arduino ploču preporučljivo je odspojiti TX i RX pinove jer se koristi serijska komunikacija.

4. Android mobilna aplikacija

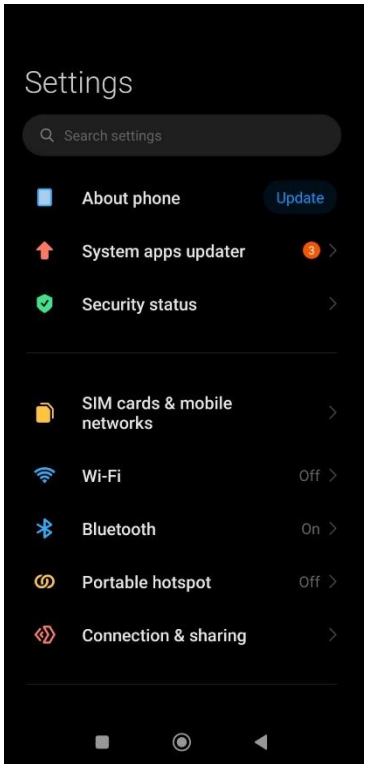
U ovom poglavlju je prikazana izrada Android mobilne aplikacije. Kao što je već ranije spomenuto Android mobilna aplikacija je pisana u jeziku Kotlin korištenjem u Android Studio IDE-u. Prije same izrade aplikacije potrebno je namjestiti opcije pametnog telefona tj. omogućiti opcije programera (engl. *developer options*) u postavkama samog pametnog telefona. U ovom završnom radu korišten je pametni telefon Xiaomi Redmi 9T.

4.1. Omogućavanje programerskih opcija

Prije omogućavanja opcije programera u pametnom telefonu, potrebno je definirati opcije programera i što nam omogućavaju.[13]

Opcija programera u pametnom telefonu predstavlja izbornik koji inicijalno nije prikazan svim korisnicima. Google je uključio sve vrste opcija za programere aplikacija kako bi bolje testirali svoje aplikacije. Krajnji korisnici mogu iskoristiti mogućnosti koje su ponuđene, kao što je: veličina zaslona, brzina animacija, zadani USB način rada, USB otklanjanje pogrešaka (engl. *USB debugging*), više prozora i mnogo toga više. Također kroz ovaj izbornik može se vidjeti korištenje RAMA (Radna memorija) pametnog telefona. Razlog zbog kojeg Google skriva postavke od svakodnevnih korisnika svojih uređaja je taj što mijenjanje nekih nepoznatih opcija može oštetiti uređaj sve dok se postavke ne vrate tvorničke.

Prvi korak je u izborniku pametnog telefona potrebno je pronaći postavke (engl. *settings*). Klikom na ikonu postavke otvara se novi prozor koji se prikazuje na slici 12.

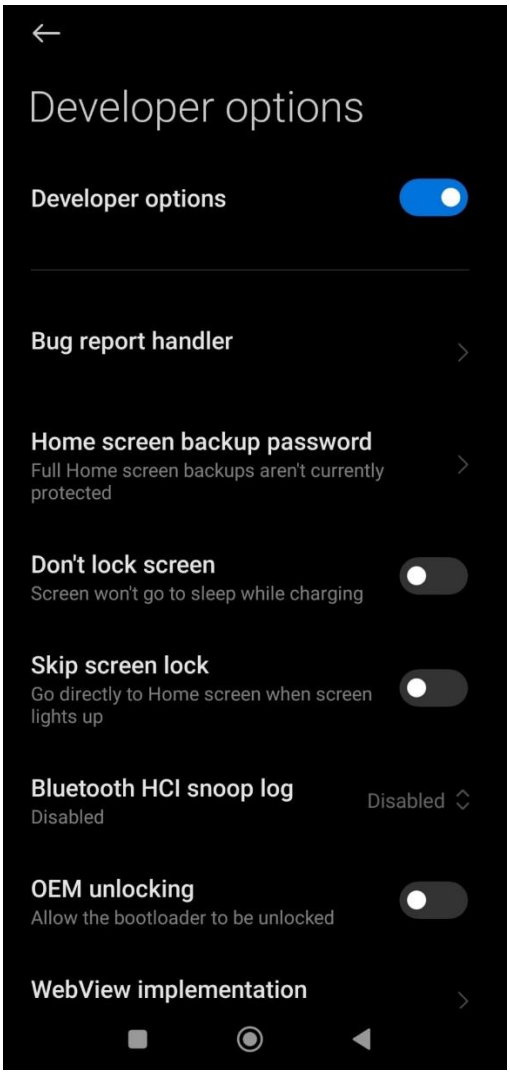


Slika 12: Postavke pametnog telefona, prvi korak

Drugi korak je klikom na ponuđenu opciju O telefonu (engl. *About phone*) otvara se novi prozor s ponuđenom opcijom sve specifikacije (engl. *All specs*) koja vodi na sljedeći prozor prikazan na slici 13.

Slika 13: Prikaz postavki pametnog telefona, drugi korak

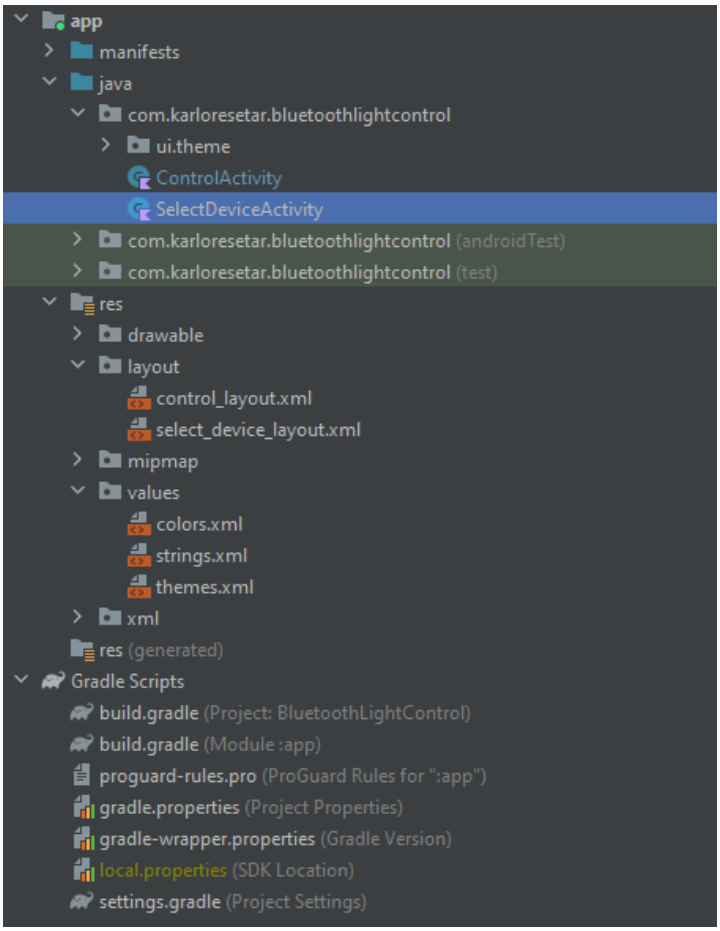
Za omogućivanje programerskih postavki potrebno je više puta brzo kliknuti na prozor MIUI version te će se pojaviti poruka na zaslonu You are now a developer! Zatim se ide na **Postavke > Dodatne postavke > Mogućnosti razvojnog programera** za pristup postavkama razvojnog programera gdje se omogućuje opcija za razvojnog programera.



Slika 14: Prozor za opcije programera

Opcija koju treba uključiti u ovome izborniku je USB debugging i Install via USB omogućavajući pametnom telefonu instaliranje aplikacija putem USB-a iz Android Studija.

4.2. Izrada Android mobilne aplikacije



Slika 15: Struktura Android mobilne aplikacije

U Android Studiju nalaze se direktoriji i datoteke koje su ključne za razvoj Android mobilne aplikacije. Direktorij `manifests` koji sadrži datoteku `AndroidManifest.xml` je datoteka koja opisuje osnovne informacije o mobilnoj aplikacija, kao što je ime paketa, komponente aplikacije (aktivnosti, usluge, prijemnici itd.), potrebne dozvole (dozvola za uključivanje bluetootha), verzija aplikacije i mnoge druge detalje vezane uz konfiguraciju same aplikacije.

`Res` direktorij ili pod punim imenom resursi (engl. *resources*) služi za pohranu svih resursa mobilne aplikacije, na primjer layout gdje se pohranjuje XML datoteke koje služe za dizajn korisničkog sučelja za određenu aktivnost. `Drawable` direktorij služi za pohranu slika i grafika koje se koristi u mobilnoj aplikaciji.

Kod `Mipmap` dijela se pohranjuju ikone koje se koriste za glavnu ikonu aplikacije na različitim razlučivostima ekrana ovisno o pametnom telefonu. `Values` je direktorij gdje se pohranjuju resursi kao što su boje, teme, stringovi, dimenzije itd. koje se koriste u aplikaciji. Sljedeći direktorij koji sadrži konfiguracijske skripte samoga projekta je `GradleScripts` u kojem se nalaze datoteke poput `build.gradle(Module)`, `build.gradle(Project)`, `settings.gradle` i ostale. Datoteka `build.gradle(Module)` definira postavke za modul aplikacije kao što je uključivanje ovisnosti (engl. *dependencies*), postavke kompilacije, plugini itd. dok datoteka `build.gradle(Project)` definira postavke za cjelokupni projekt, kao što su verzije alata (engl. *tool versions*), globalne ovisnosti i druge postavke koje se odnose na cijeli projekt. Datoteka `settings.gradle` definira konfiguraciju modula unutar projekta i njihove putanje.

Android mobilna aplikacija je podijeljena u dva dijela to jest dvije aktivnosti. Prva aktivnost je `SelectDeviceActivity` koja služi za ispis svih dostupnih bluetooth uređaja te za osposobljavanje veze u ovom slučaju Arduina i Android mobilne aplikacije preko Bluetooth veze.

Druga aktivnost je `ControlActivity` gdje je napisana logika za upravljanje LED diode te osposobljavanje svih značajki izrađene putem layout.xml datoteke kao što je botun za uključivanje/isključivanje, treptanje LED diode ili za kontroliranje jačine svjetlosti LED diode.

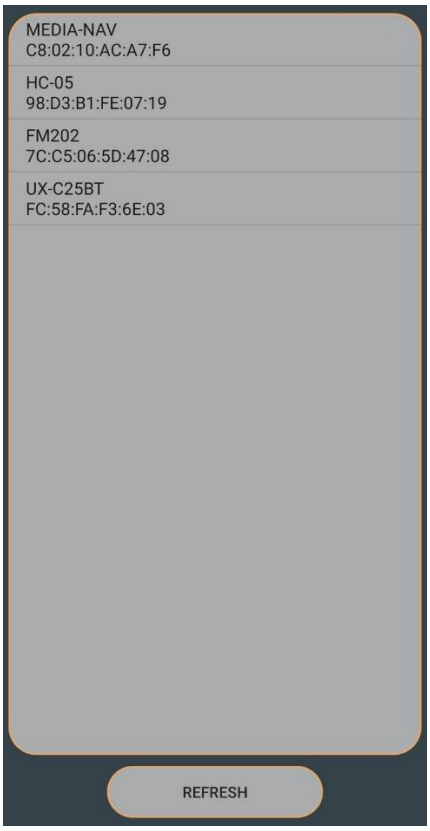
4.3. Definicija korisničkog sučelja za prikaz uparenih Bluetooth uređaja

Datoteka `Select_device_layout.xml` je dizajn izgleda korisničkog sučelja koja služi za prikazivanje liste dostupnih bluetooth uređaja. Koristi tradicionalni pristup izgleda temeljen na XML-u s ograničenim izgledom (engl. *ConstraintLayout*) koji dolazi iz `AndroidX` biblioteke za definiranje korisničkog sučelja za Android aktivnosti, u ovom slučaju `SelectDeviceActivity`.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
    xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#35424a"

tools:context="com.karloresetar.bluetoothlightcontrol.SelectDeviceActi
vity">

    <ListView
        android:id="@+id/select_device_list"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginTop="10dp"
        android:layout_marginBottom="10dp"
        android:layout_marginLeft="5dp"
        android:layout_marginRight="5dp"
        android:background="@drawable/layout_radius"

app:layout_constraintBottom_toTopOf="@id/select_device_refresh"
        app:layout_constraintTop_toTopOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

    <Button
        android:id="@+id/select_device_refresh"
        android:layout_width="200dp"
        android:layout_height="wrap_content"
        android:layout_marginStart="20dp"
        android:layout_marginEnd="20dp"
        android:layout_marginBottom="20dp"
        android:text="@string/refresh"
        android:textColor="@color/btn_color"
        android:background="@drawable/layout_radius"
        app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayout>
```

Ispis 5: XML kôd korisničkog sučelja za ispis i odabiranje dostupnih uređaja

Linija kôda `tools:context` se koristi za određivanje konteksta, Unutar `ContraintLayout` postoje dvije komponente, jedna od njih je `ListView` gdje je dodijeljen `id/select_device_list` koji će biti potreban za prikazivanje liste svih dostupnih bluetooth uređaja te botun koji služi za osvježavanje liste.



Slika 16: Izgled korisničkog sučelja za ispis i osvježavanje liste uređaja

4.4. Definicija korisničkog sučelja za upravljanje LED-icom

Izgled za `control_layout.xml` je napravljen tako da se preko kontrolne aktivnosti mogu slati podaci prema Arduinu pomoću bluetooth modula. U ovome izgledu napravljeni su botuni za uključivanje/isključivanje, treptanje, kontroliranje jačine svjetlosti LED diode pomoću klizača te pomoću unosa jačine svjetlosti u `EditText` koji je standardni widget to jest polje za unos podataka u android aplikacijama. Također napravljen je botun za prekidanje bluetooth veze.

```
<?xml version="1.0" encoding="utf-8"?>
<androidx.constraintlayout.widget.ConstraintLayout
xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:app="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:background="#35424a"

tools:context="com.karloresetar.bluetoothlightcontrol.ControlActivity"
>

    <androidx.appcompat.widget.SwitchCompat
        android:id="@+id/control_led_switch"
        style="@style/SwitchTextStyle"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_margin="25dp"
        android:textOff="@string/text_off"
        android:textOn="@string/text_on"
        android:thumb="@drawable/thumb"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        app:showText="true"
        app:track="@drawable/track" />


    <Button
        android:id="@+id/control_led_blink"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="16dp"
        android:text="@string/blink"
        android:background="@drawable/custom_buttons"
        android:textColor="@color/btn_color"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/control_led_switch"
/>
```

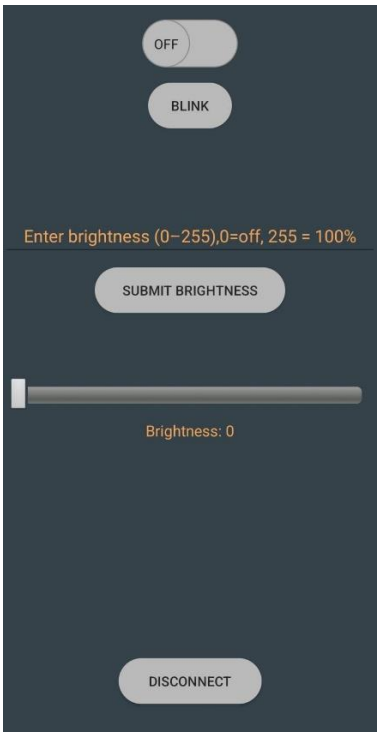
Ispis 6: XML kôd izgleda za kontroliranje LED diode botunima

Kod ispisa 6 može se uočiti korištenje widgeta `SwitchCompat` [14] koji je zapravo botun i radi na način kao prekidač s dva stanja te se može birati između dvije opcije uključi/isključi. Koristi se tako da korisnik kontrolira botun naprijed-natrag kako bi odabrao opciju uključivanja/isključivanja LED diode ili jednostavno dodirnuti botun za prebacivanje opcija. `SwitchCompat` je verzija Switch widgeta koji radi na uređajima od verzije API-a 7 pa nadalje.

Osim polja za unos podataka jačine svjetlosti također postoji značajka za kontroliranje jačine svjetlosti LED diode korištenjem klizača. Kôd za klizač koji je napravljen pomoću Android SeekBara [15]. SeekBar je ekstenzija trake za napredak (engl. *progress bar*) koja omogućuje korištenje botuna na progress baru. SeekBar je jedan od važnih elemenata korisničkog sučelja koji pruža opciju odabira cjelobrojnih vrijednosti unutar definiranog raspona poput 0 do 255. U ispisu 7 može se vidjeti XML kôd za progress bar.

```
<SeekBar
        android:id="@+id/brightness_slider"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="18dp"
        android:max="255"
        android:paddingTop="50dp"
        android:progress="0"

android:progressDrawable="@android:drawable/progress_horizontal"
        android:thumb="@android:drawable/btn_default_small"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@id/submit_button" />
```

Ispis 7: XML kôd progress bar za kontroliranje jačine svjetlosti LED diode



Slika 17: Izgled korisničkog sučelja za upravljanje LED diode

4.5. Funkcionalnost kôda android mobilne aplikacije

U ovome dijelu će biti objašnjen najvažniji dijelovi kôda te funkcionalnosti dviju aktivnosti. Prva aktivnost koja je potrebna da bi mobilna aplikacija radila je `SelectDeviceActivity`, koja omogućuje spajanje mobilne aplikacije s Arduino pločom preko Bluetooth veze.

4.5.1. Spajanje android mobilne aplikacije bluetooth vezom

Jedan od najbitnijih dijelova prije pisanja kôda za spajanje na Arduino ploču preko Bluetooth HC-05 modula su dozvole (engl. *permissions*) koje se deklariraju u datoteci `AndroidManifest.xml`.[16]

```
<uses-permission android:name="android.permission.BLUETOOTH" />
<uses-permission android:name="android.permission.BLUETOOTH_CONNECT"
/>
<uses-permission android:name="android.permission.BLUETOOTH_ADMIN" />
```

Ispis 8: Dozvole za korištenje Bluetooth funkcionalnosti u Android aplikaciji

`BLUETOOTH` dozvola nudi mogućnost komunikacije s uređajima putem Bluetootha. Ovo je osnovna dozvola potrebna za rad s Bluetooth funkcionalnošću.

`BLUETOOTH_CONNECT` dozvola nudi mogućnost aplikaciji uspostavu Bluetooth veze s uparenim uređajima.

`BLUETOOTH_ADMIN` je dozvola koja nudi mogućnost to jest omogućuje aplikaciji administratorska prava za kontrolu nad Bluetooth funkcionalnošću na uređaju. Omogućuje aplikaciji uključivanje/isključivanje Bluetootha. Također omogućuje upravljanje vezama i izvođenje drugim administratorskih zadataka vezanih za Bluetooth funkcionalnosti. U ispisu 9 prikazan je način korištenja Bluetooth dozvole za provjeru pristupa Bluetootha od strane mobilne aplikacije.

```
private fun checkBluetoothPermission() {
    if (ActivityCompat.checkSelfPermission(
            this,
            Manifest.permission.BLUETOOTH
        ) != PackageManager.PERMISSION_GRANTED
    ) {
        ActivityCompat.requestPermissions(
            this,
            arrayOf(Manifest.permission.BLUETOOTH),
            PERMISSION_REQUEST_CODE
        )
    } else {
        pairedDeviceList()
    }
}
```

Ispis 9: Kotlin kôd za dopuštenje pristupa Bluetooth funkcionalnosti

Ovaj blok kôda radi tako da provjerava je li aplikacija dobila potrebno dopuštenje za pristup Bluetooth funkcionalnosti na uređaju. Ako dopuštenje nije dodijeljeno, funkcija će zatražiti korisnika da odobri dopuštenje, obratno ako je dopuštenje već dodijeljeno od strane korisnika funkcija će pozvati funkciju `pairedDeviceList()`, u ispisu 10 prikazan je kôd funkcije `pairedDeviceList()`.

```
Private fun pairedDeviceList(){

m_pairedDevices = bluetoothAdapter!!.bondedDevices
    val list: ArrayList<BluetoothDeviceInfo> = ArrayList()

    if (!m_pairedDevices.isEmpty()) {
        for (device: BluetoothDevice in m_pairedDevices) {
            val name = device.name ?: "Unknown Device"
            val address = device.address
            val deviceInfo = BluetoothDeviceInfo(name, address)
            list.add(deviceInfo)
            Log.i("device", deviceInfo.toString())
        }
    } else {
        showToast(this, "No paired Bluetooth devices found")
        if (!bluetoothAdapter!!.isEnabled) {
            val enableBtIntent =
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE)
            startActivityForResult(enableBtIntent,
REQUEST_ENABLE_BLUETOOTH)
        } else {
            checkBluetoothPermission()
        }
    }

    val adapter = ArrayAdapter(this,
android.R.layout.simple_list_item_1, list)
    val deviceList: ListView = findViewById(R.id.select_device_list)
    deviceList.adapter = adapter
    deviceList.onItemClickListener = AdapterView.OnItemClickListener {
_, _, position, _ ->
        val device: BluetoothDeviceInfo = list[position]
        val address: String = device.address

        val intent = Intent(this, ControlActivity::class.java)
        intent.putExtra(EXTRA_ADDRESS, address)
        startActivity(intent)
    }
}
```

Ispis 10: Funkcija pairedDeviceList()

Funkcija `pairedDeviceList()` služi za prikazivanje uparenih uređaja te omogućuje korisniku da se spoji na jedan od uparenih uređaja, te preusmjeruje korisnika na aktivnost za kontroliranje LED diode. Ako uparenih uređaja nema na listi to jest ako nisu pronađeni prikazana je poruka. Ako Bluetooth nije uključen, aplikacija u tom slučaju traži dopuštenje od korisnika za uključivanje Bluetooth veze s mobilnog uređaja.

Također je važno napomenuti dvije klase koje su potrebne u ovome završnom radu koje se koriste za upravljanje bežičnom Bluetooth komunikacijom prema uređaju, a to su `BluetoothManager` i `BluetoothAdapter`.

`BluetoothManager` je klasa koja omogućava upravljanje Bluetooth uređajima i funkcionalnostima na Android uređaju. Pruža metode za traženje bluetooth uređaja, uparivanje i povezivanje s uređajima te za upravljanje Bluetooth servisima.

`BluetoothAdapter` je instanca `BluetoothManagera` koja se koristi za komunikaciju između dva uređaja. `BluetoothAdapter` također nudi mogućnost za upravljanje komunikacijom s drugim uređajem poput slanja i primanja podataka preko Bluetootha. U ispisu 11 prikazan je kôd korištenja varijabli `bluetoothManager` i `bluetoothAdapter` koje u metodi `OnCreate` provjeravaju je li uređaj podržava Bluetooth, ako Bluetooth nije omogućen poziva se `checkBluetoothPermission` funkcija za omogućavanje Bluetootha u suprotnom poziva se funkcija `pairedDeviceList` ako je Bluetooth već omogućen.

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.select_device_layout)

    bluetoothManager = getSystemService(Context.BLUETOOTH_SERVICE) as
BluetoothManager
    bluetoothAdapter = bluetoothManager.adapter

    if (bluetoothAdapter == null) {
        showToast(this,"Device does not support Bluetooth")
        return
    }
    if (!bluetoothAdapter!!.isEnabled) {
        val enableBtIntent =
Intent(BluetoothAdapter.ACTION_REQUEST_ENABLE)
        startActivityForResult(enableBtIntent,
REQUEST_ENABLE_BLUETOOTH)
    } else {
        checkBluetoothPermission()
    }
    val refreshButton: Button =
findViewById(R.id.select_device_refresh)
    refreshButton.setOnClickListener { pairedDeviceList() }
}
```

Ispis 11: Kôd za provjeru Bluetooth veze

4.5.2. Kontroliranje LED diode android mobilnom aplikacijom

Nakon uspješnog spajanja mobilne aplikacije s Arduinom preko bluetooth veze, aplikacija preusmjerava korisnika na sljedeću aktivnost `ControlActivity`, gdje se nalaze značajke za kontroliranje LED diode. U ispisu 12 prikazane su jedne od deklariranih varijabli korištene u spomenutoj aktivnosti.

```
class ControlActivity : AppCompatActivity() {

companion object {
    var m_myUUID: UUID = UUID.fromString("00001101-0000-1000-8000-
00805F9B34FB")
    var m_bluetoothSocket: BluetoothSocket? = null
    lateinit var m_progress: ProgressDialog
    lateinit var m_bluetoothAdapter: BluetoothAdapter
    var m_isConnected: Boolean = false
    lateinit var m_address: String
}

lateinit var brightnessInput: EditText
lateinit var submitButton: Button
lateinit var brightnessSlider: SeekBar
lateinit var brightnessValueText: TextView
lateinit var controlLedSwitch: SwitchCompat
```

Ispis 12: Deklaracija varijabli u aktivnosti za kontrolu LED diode

Korištene varijable koje pripadaju unutar bloka `companion object` [17] su statički članovi klase koje se koriste za pohranjivanje različitih vrsta podataka koji su potrebni za upravljanje Bluetooth komunikacijom i interakcijom unutar same mobilne aplikacije. Varijable koje su deklarirane ispod bloka `companion object`, su varijable u kojima će se pohranjivati reference na elemente iz korisničkog sučelja to jest to su elementi iz korisničkog sučelja `control_layout.xml` kako bi aplikacija mogla komunicirati s njima putem kôda.

`m_myUUID` je varijabla kojoj je dodijeljena određena UUID (Univerzalni jedinstveni identifikator) vrijednost pomoću metode `UUID.fromString` koja se koristi za uspostavu Bluetooth veze.

`m_bluetoothSocket` je varijabla koja se koristi za pohranu instance `BluetoothSocket` klase, koja služi za komunikaciju između Bluetooth uređaja, uspostavljanje veze te za čitanje i pisanje podataka između uređaja.

U ispisu 13 prikazan je način referenciranja elemenata iz korisničkog sučelja, korištenjem metode `findViewById` koja se koristi za pronalaženje i dobivanje reference na element iz korisničkog sučelja `control_layout.xml` kako bi se moglo omogućiti funkcionalno upravljanje to jest interakcija komponenti u mobilnoj aplikaciji.

```
controlLedSwitch = findViewById(R.id.control_led_switch)

val controlLedBlinkButton: Button =
findViewById(R.id.control_led_blink)

val controlLedDisconnectButton: Button =
findViewById(R.id.control_led_disconnect)

brightnessInput = findViewById(R.id.brightness_input)

submitButton = findViewById(R.id.submit_button)

brightnessSlider = findViewById(R.id.brightness_slider)

brightnessValueText = findViewById(R.id.brightness_value_text)
```

Ispis 13: Referenciranje elemenata iz control_layout.xml korisničkog sučelja

Za komunikaciju Android mobilne aplikacije s Arduino Uno pločom korištene su funkcije u kojima se šalju podaci putem JSON formata preko Bluetooth veze koristeći `BluetoothSocket`.

```
private fun sendCommand(command: String, value: Int? = null) {
        if (m_bluetoothSocket != null) {
            try {
                val json = JSONObject()
                json.put("command", command)
                value?.let { json.put("value", it) }

m_bluetoothSocket!!.outputStream.write(json.toString().toByteArray())
            } catch (e: IOException) {
                e.printStackTrace()
            }
        }
    }
```

Ispis 14: sendCommand() funkcija za slanje podataka tipa Integer

`sendCommand()` funkcija se koristi za slanje podataka preko Bluetooth veze. Prima dva argumenta, prvi argument je varijabla command tipa String to jest command je identifikator preko kojeg će Arduino Uno ploča moći raspoznati primljeni podatak. Razlog korištenja identifikatora to jest „ključa“ je ta što se koristi JSON objekt koji sadrži svoj ključ (u ovome slučaju *command*) i sadrži value (engl. *vrijednost*) tipa Integer. Ako `m_bluetoothSocket` nije null, onda se pomoću `'JSONObject'` stvara JSON objekt koji sadrži opisane parametre te se šalje preko Bluetooth veze. Pomoću `sendCommand()` funkcije kontrolira se uključivanje, isključivanje, treptanje i jačina svjetlosti LED diode korištenjem klizača koja ima različiti key pod imenom „set_brightness“.

Funkcija `sendBrightnessCommand()` kontrolira jačinu svjetlosti LED diode pomoću polja za unos podataka (u ovom slučaju 0-255). Radi tako da se iz teksta unesenog u 'brightnessInput' polje čita String koji označava željenu jačinu svjetlosti. Ako polje za unos podataka nije prazno, podatak se pokuša pretvoriti u tip Integer. Ako je pretvorba prošla uredno i broj je između 0 i 255, poziva se funkcija `sendCommand()` s prvim argumentom „set_brightness“ i drugim argumentom koji je vrijednost jačine svjetlosti.

```
private fun sendBrightnessCommand() {
        val brightnessText = brightnessInput.text.toString()
        if (brightnessText.isNotEmpty()) {
            val brightnessValue = brightnessText.toIntOrNull()
            if (brightnessValue != null && brightnessValue >= 0 &&
brightnessValue <= 255) {
                sendCommand("set_brightness", brightnessValue)
            } else {
                showToast("Invalid input; must be between 0-255!")
            }
        } else {
            showToast("Please enter a brightness value")
        }
    }
```

Ispis 15: sendBrightnessCommand() funkcija za kontrolu diode

U ispisu 16 prikazano je korištenje funkcija `sendBrightnessCommand` i `sendCommand` za komunikaciju s Arduino Uno pločom.

```
controlLedSwitch = findViewById(R.id.control_led_switch)
            controlLedSwitch.setOnCheckedChangeListener { _, isChecked
->
                if (isChecked) {
                    sendCommand("turn_on")
                } else {
                    sendCommand("turn_off")
                }
            }

            val controlLedBlinkButton: Button =
findViewById(R.id.control_led_blink)
            controlLedBlinkButton.setOnClickListener {
sendCommand("blink") }

            val controlLedDisconnectButton: Button =
findViewById(R.id.control_led_disconnect)
            controlLedDisconnectButton.setOnClickListener {
disconnect() }

            brightnessInput = findViewById(R.id.brightness_input)
            submitButton = findViewById(R.id.submit_button)
            submitButton.setOnClickListener { sendBrightnessCommand()
}

            brightnessSlider = findViewById(R.id.brightness_slider)
            brightnessValueText =
findViewById(R.id.brightness_value_text)
            brightnessSlider.setOnSeekBarChangeListener(object :
SeekBar.OnSeekBarChangeListener {
                override fun onProgressChanged(seekBar: SeekBar?,
progress: Int, fromUser: Boolean) {
                    val brightnessText =
getString(R.string.brightnessProgress) + " $progress"
                    brightnessValueText.text = brightnessText
                    sendCommand("set_brightness", progress)


                }

                override fun onStartTrackingTouch(seekBar: SeekBar?)
{}
                override fun onStopTrackingTouch(seekBar: SeekBar?) {}
            })
        }
```

Ispis 16: Uporaba funkcija za komunikaciju s Arduino Uno pločom

Posljednja funkcija koja je vrijedna spomena je `connectToDevice()`. Funkcija koja omogućava povezivanje mobilne aplikacije s Bluetooth modulom da bi se mogle koristiti značajke za upravljanje LED diode. Funkcija se sastoji od prozora koji prikazuje poruku za spajanje, dijela u kojem se provodi pokušaj povezivanja s Bluetooth modulom gdje se provjerava `bluetoothSocket` je li je njegova veza već postavljena ili uspostavljena. Ako `bluetoothSocket` nije postavljen ili je prethodna veza nepostojeća dohvaća se instanca `bluetoothAdaptera` i traži se uređaj na temelju adrese `'m_address'`. `BluetoothSocket` se stvara pomoću funkcije `'createInsecureRfcommSocketToServiceRecord(m_myUUID)'` te se pokušava uspostaviti veza pozivom metode `.connect()` na socketu. U ispisu 17 nalazi se opisani dio kôda.

```
private fun connectToDevice() {
        m_progress = ProgressDialog.show(this, "Connecting...",
"please wait")
        lifecycleScope.launch(Dispatchers.IO) {
            var connectSuccess = true
            try {
                if (m_bluetoothSocket == null || !m_isConnected) {
                    m_bluetoothAdapter =
BluetoothAdapter.getDefaultAdapter()
                    val device: BluetoothDevice =
m_bluetoothAdapter.getRemoteDevice(m_address)
                    m_bluetoothSocket =
device.createInsecureRfcommSocketToServiceRecord(m_myUUID)
                    m_bluetoothSocket!!.connect()
                }
            } catch (e: IOException) {
                connectSuccess = false
                e.printStackTrace()
            }

            launch(Dispatchers.Main) {
                m_progress.dismiss()
                if (connectSuccess) {
                    val deviceName =
m_bluetoothSocket!!.remoteDevice.name
                    showToast("Connected to device: $deviceName")
                } else {
                    showToast("Could not connect to device")
                    finish()
                }
                m_isConnected = connectSuccess
            }
        }
    }
```

Ispis 17: Funkcija za uspostavljanje veze s Bluetooth uređajem

5. Zaključak

U ovom završnom radu je napravljena Android mobilna aplikacija u jeziku Kotlin koja omogućuje upravljanje LED diode putem pametnog telefona i ploče Arduino Uno koristeći Bluetooth vezu pomoću HC-05 modula. Android mobilna aplikacija omogućuje spajanje na bluetooth uređaj s pametnog telefona, te upravljanje LED diodom gdje je omogućeno uključivanje, isključivanje, treptanje i prilagodba jačine svjetlosti. Ova mobilna aplikacija je posve prilagođena korisniku te omogućuje jednostavno korištenje same aplikacije.

Pisanje rada poput ovog može doprinijeti sposobnosti razumijevanja raznih tehnologija te njihovu primjenu, gdje se može steći solidno iskustvo i znanje u radu s Arduino platformom, Bluetooth tehnologijom i razvojem Android aplikacija. Izrađenu mobilnu aplikaciju je zasigurno moguće dodatno poboljšati. Neka od tih poboljšanja mogu biti dodatne postavke, primjerice sinkronizacija LED svjetla s glazbom, poboljšanje korisničkog sučelja dodavanjem animacija, proširenje na druge mobilne platforme kao što su iOS uređaji i mnoge druge.

Konačno, može se zaključiti kako je završni rad uspješno prikazao razvoj i korištenje Android mobilne aplikacije na pametnom telefonu za komunikaciju s Arduino Uno pločom putem Bluetooth tehnologije.

Literatura

[1] M. Li, »Why Bluetooth IoT?,« 3 Studeni 2020. [Mrežno]. Available: https://www.mokoblue.com/why-bluetooth-iot/. [Posjećeno 1. 8. 2023.].

[2] Javatpoint, »Arduino IDE,« [Mrežno]. Available: https://www.javatpoint.com/arduino-ide. [Posjećeno 1. 8. 2023.].

[3] »Arduino dokumentacija,« [Mrežno]. Available: https://docs.arduino.cc/learn/. [Posjećeno 1. 7. 2023.].

[4] »Android Studio Screen,« [Mrežno]. Available: https://javatutorial.net/install-configure-android-studio/. [Posjećeno 2. 8. 2023.].

[5] »Android dokumentacija,« [Mrežno]. Available: https://developer.android.com/docs. [Posjećeno 1. 7. 2023.].

[6] »Kotlin docs,« [Mrežno]. Available: https://kotlinlang.org/docs/home.html. [Posjećeno 1. 7. 2023.].

[7] »Kotlin vs Java: Which will Succeed Android Development in Coming Times?,« [Mrežno]. Available: https://www.vervelogic.com/blog/kotlin-vs-java/. [Posjećeno 2. 8. 2023.].

[8] Makerspaces, »Arduino For Beginners,« [Mrežno]. Available: https://www.makerspaces.com/arduino-uno-tutorial-beginners/. [Posjećeno 3. 8. 2023.].

[9] T. Hirzel, »Basics of PWM (Pulse Width Modulation),« [Mrežno]. Available: https://docs.arduino.cc/learn/microcontrollers/analog-output. [Posjećeno 2. 7. 2023.].

[10] »HC-05 - Bluetooth Module,« 16. 7. 2021. [Mrežno]. Available: https://components101.com/wireless/hc-05-bluetooth-module. [Posjećeno 3. 8. 2023.].

[11] [Mrežno]. Available: https://mechstuff.com/control-leds-with-your-android-arduino-bluetooth-module-tutorial/. [Posjećeno 3. 8. 2023].

[12] »ArduinoJson Documentation,« [Mrežno]. Available: https://arduinojson.org/v6/doc/. [Posjećeno 12. 8. 2023].

[13] E. Bardakçı, »How to Enable Developer Options on Xiaomi Devices,« 31. 12. 2021. [Mrežno]. Available: https://xiaomiui.net/how-to-enable-developer-options-on-xiaomi-devices-2504/. [Posjećeno 1. 7. 2023].

[14] »SwitchCompat,« [Mrežno]. Available: https://developer.android.com/reference/kotlin/androidx/appcompat/widget/SwitchCompat. [Posjećeno 24. 7. 2023].

[15] Praveenruhil, »SeekBar in Kotlin,« 28. 3. 2022. [Mrežno]. Available: https://www.geeksforgeeks.org/seekbar-in-kotlin/. [Posjećeno 20. 7. 2023].

[16] »Bluetooth overview,« [Mrežno]. Available: https://developer.android.com/guide/topics/connectivity/bluetooth. [Posjećeno 1. 7. 2023].

[17] B. Khani, »Kotlin Companion Object,« 29. 9. 2022. [Mrežno]. Available: https://www.baeldung.com/kotlin/companion-object. [Posjećeno 1. 7. 2023].