Šolski center Novo mesto

Srednja elektro šola in tehniška gimnazija

Šegova ulica 112

8000 Novo mesto

Maturitetna seminarska naloga pri predmetu računalništvo

APLIKACIJE IN INFORMACIJSKI SISTEMI – MORSE PREVAJALNIK

Šolsko leto 2020/21

Avtor: Dejan Jarc, T4C

Mentor: dr. Albert Zorko, univ. dipl. inž.

Trebnje, januar 2021

POVZETEK

Mobilne aplikacije so v sodobnem času nepogrešljive. Olajšujejo nam življenja in nam poenostavijo delo, hkrati pa nam lahko služijo kot učni pripomočki. V tej seminarski nalogi bom predstavil potek izdelave in delovanje moje mobilne aplikacije Morse prevajalnik, ki omogoča prevajanje med navadnim besedilom in Morse kodo ter služi kot učilo. Dokazal bom pomembnost in koristnost mobilnih učil za demonstracijo in lažje učenje na primeru učila za telegrafijo oziroma Morsejevo abecedo. Znanje telegrafije je kljub napredku v komunikacijskih tehnologijah zelo koristno. S pomočjo literature, internih gradiv in znanja o programskih orodjih bom v urejevalnem okolju Android Studio ustvaril aplikacijo, ki pripomore k učenju Morsejeve abecede s funkcijo prevajanja. Aplikacija bo narejena z namenom uporabe pri poučevanju telegrafije otrokom in tistim, ki si želijo poenostaviti delo v telegrafiji ali preverjati pravilnost prevodov.

Ključne besede: aplikacija, Morse, prevajalnik, Android, Java, metoda, Android Studio

ABSTRACT

Mobile applications are indispensable in the present time. They make our lives easier and simplify our jobs, and at the same time they can be used as learning aids. In this paper, I will demonstrate the development and operation of my mobile application Morse Translator, which allows translation between plain text and Morse code and serves as a learning aid. I will prove the importance and usefulness of mobile teaching aids for demonstration and easier learning in the case of a telegraphy ,or the Morse code, teaching aid. Knowledge of telegraphy is very useful despite advances in communication technologies. With the help of literature, internal materials and knowledge of software tools, I will create an application in the Android Studio editing environment that helps to learn the Morse code with the translation function. The application will be made for the purpose of teaching telegraphy to children and those who want to simplify their work in telegraphy or to check the correctness of translations.

Key words: application, Morse, translator, Android, Java, method, Android Studio

KAZALO VSEBINE

KAZALO SLIK

1. UVOD

Telegrafija je način komuniciranja na daljavo s pomočjo dogovorjene tehnike, ki se imenuje Morsejeva abeceda. **Morsejeva abeceda**[1] je oblika komuniciranja, kjer pošiljatelj in prejemnik komunicirata s pomočjo kode, ki temelji na dolgih in kratkih signalih. Kombinacije le-teh se uporabijo, da se šifrira besedilo v različne oblike (električni signal, svetlobni signal ali zvočni signal). (1) (2)

V preteklosti je to omogočalo komunikacijo zaradi nerazvite tehnologije za prenos zvoka, danes pa se uporablja v vojski, mornarici, radioamaterstvu, taborništvu ali kot hobi. (1) Ker je uporaba še vedno aktualna in uporabna, sem se odločil v programskem jeziku Java s pomočjo okolja Android Studio narediti aplikacijo, ki služi kot hiter prevajalnik za pošiljatelja ali prejemnika. Imenuje se **Morse prevajalnik**. V aplikaciji bom omogočil vnos besedila, katerega z gumbi prevedemo ali iz besedila v Morse kodo ali iz Morse kode v besedilo. Poleg tega bom omogočil še nekaj funkcij za lažjo uporabo kot je gumb za brisanje besedila ali gumb za kopiranje prevedenega besedila. Po obliki bo aplikacija sestavljena iz besedilnih polj in gumbov, ki ali prevajajo in manipulirajo z besedilo ali odprejo navodila in pomoč uporabniku.

Namen aplikacije Morse prevajalnik je postati učilo za učenje telegrafije in preverjanje pravilnosti prevedenih besedil. Z njeno uporabo želim poenostaviti učenje telegrafije in zmanjšati učno krivuljo telegrafije na nivo, kjer je njena uporaba in učenje preprosto za otroke. Aplikacijo bom zasnoval tako, da bom lahko uporabo implementiral v delovanje pri taborniških aktivnostih v domačem kraju in olajšal delo vodnikov, hkrati pa vsem predstavil preprostost in uporabnost Morsejeve abecede.

1.1. ANDROID STUDIO

V programskem okolju **Android Studio** je Java, poleg Kotlin in C++, glavni programski jezik, s katerim lahko razvijamo aplikacije za Android. Android Studio je uradno integrirano razvijalno okolje za aplikacije Android, ki temelji na IntelliJ IDEA[2] in omogoča pregledno programiranje, uporaben pa je tudi vgrajen emulator za testiranje. (3)

Android Studio ima svoje knjižnice, ki so koristne pri delu z vmesniškimi elementi aplikacije (angl. Widget), na primer gumbi in polji besedila. Vsak element ima svoj razred, ki je shranjen v Androidovih knjižnicah posebej. (3) Android Studio podobno kot Eclipse predlaga popravke in izboljšave v obliki skrajšav kode v bolj pregledne oblike. Zelo uporabno je tudi njegovo sprotno opozarjanje na sintaktične napake.

[1] Kodiranje črk, številk in znakov s pomočjo kombinacije dolgih in kratkih znakov (npr. črte in pike)

[2] Integrirano razvijalno okolje napisano v Javi, ki ga je razvilo podjetje JetBrains za namen razvijanja kode v različnih programskih jezikih

1.2. POMEMBNE JAVA METODE V ANDROID STUDIO

Najbolj koristne so vgrajene metode, ki omogočajo posreden dostop med elementi in kodo, ki jih kliče. Pri izdelavi bodo najbolj priročne:

1.) **SetOnClickListener**:

Funkcija omogoča vezavo vmesnika za poslušanje oz. `OnClickListener()` na vmesniški element, npr. gumb (angl. Button), kot vidimo na Sliki 1. Gnezdena metoda `onClick()` nato ob pritisku na določeni vmesniški element izvede kodo, ki je znotraj metode. (4) To funkcijo nameravam uporabiti največkrat, ker bom imel veliko dela z odzivi na pritiske in drugačne interakcije z vmesniki.

```
gumb.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        //koda, ki se izvede ob kliku je tukaj
    }
});
```

Slika 1: Dolga oblika metode »SetOnClickListener()«

V nalogi bom dolgo obliko funkcije, kakor je prikazana zgoraj, preoblikoval v krajšo obliko ter s tem strnil kodo in izboljšal preglednost in olajšal dodajanje kode v funkcijo. Primer take oblike vidimo na Sliki 2. Delovanje funkcije je popolnoma enako, je pa bolj pregledno in zato boljše.

```
gumb.setOnClickListener(v -> {
        //koda, ki se izvede ob kliku je tukaj
});
```

Slika 2: Kratka oblika metode »SetOnClickListener()«

2.) SetContentView

Funkcija omogoča določitev željene postavitve (angl. Layout) ob določenem trenutku in uporabi poljubno XML datoteko, ki predstavlja postavitev in izgled. (5) Primer oblike vidimo na Sliki 3.

```
setContentView(R.layout.postavitev);
```

Slika 3: Oblika metode »SetContentView()«

3.) **SetText** in **GetText**

Funkciji omogočata delo z besedilom v vmesniškem elementu za besedilo (angl. TextView/EditText[3] widget). (6) Ti dve metodi bom najbolj uporabljal v kombinaciji s `SetOnClickListener()`, kjer metodi uporabim za upravljanje z besedilom preko metod za prevajanje ali za premikom besedil. Metoda `setText()` elementu določi neko besedilo, `getText()` pa z elementa vzame besedilo za nadaljnjo uporabo. (6) Primer oblike vidimo na Sliki 4.

[3] **TextView** predstavlja polje z besedilom, **EditText** pa predstavlja polje, kamor lahko vnašamo besedilo.

```
String besedilo1 = VnosnoPolje.getText().toString();
String besedilo2;
Besedilo2.setText(MetodaZaPrevod(besedilo1));
```

Slika 4: Primer uporabe metod »GetText()« in »SetText()«

4.) FindViewById

Funkcija je namenjena identifikaciji vmesniškega elementa, s katerim delamo iz poljubne XML datoteke, ki predstavlja trenutno postavitev. Vsak element ima neko identifikacijsko ime (ID), preko katerega s funkcijo `findViewById()` element najdemo in z njim delamo. (5) Ko je element definiran s funkcijo `findViewById()`, je pripravljen na uporabo z Javo. Primer uporabe vidimo na Sliki 5.

```
Button gumb;
gumb = findViewById(R.id.gumb_id);
 //element je pripravljen za delo, npr. za določitev SetOnClickListener
```

Slika 5: Primer deklaracije z metodo »FindViewById()«

1.3. XML V ANDROID STUDIO

XML (Extensible Markup Language) je razširljivi označevalni jezik in je definiran kot nabor pravil za definiranje semantičnih oznak, ki dokument razdelijo na dele in le-te identificira. Ker je metaoznačevalni jezik, je sintaksa, ki je uporabljena za definiranje označb, odvisna od želene uporabe in ni omejena na določeno število označb kakor npr. HTML (Hypertext Markup Language). (7) (8)

V Android Studio je veliko označb narejenih ravno za vmesniške elemente v tem okolju. XML sicer omogoča poljubno izdelavo označb za zelo specifične potrebe, ampak za izdelavo aplikacije jih je veliko že narejenih znotraj Android Studio. To pride prav zaradi boljše preglednosti in lažje organizacije kode, kjer je pomen in namen označbe popolnoma jasen in nedvoumen. Primer XML označbe v Android Studio je vmesniški element gumb (angl. Button). Zanj je dodeljenih tudi nešteto atributov, kot vidimo na Sliki 6. Element ima zgolj eno funkcijo in to je služiti za pritiskanje oz. interakcijo ter izvedbo vezane reakcije v kodi. Podobno imajo določene naloge in lastnosti tudi drugi elementi, ki so definirani v XML.

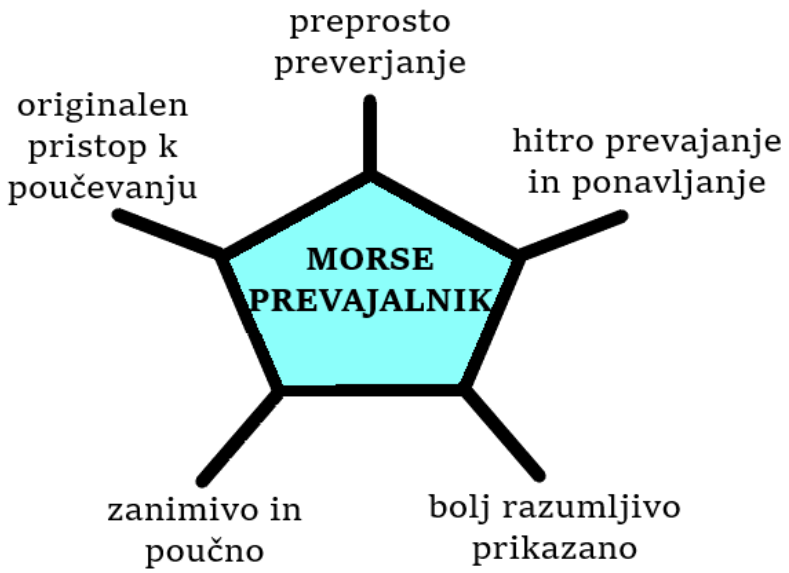
```
<Button
    android:id="@+id/gumb"
    android:layout_width="wrap_content"
    android:layout_height="45dp"
    android:layout_marginStart="15dp"
    android:layout_weight="1"
    android:text="@string/gumbBesedilo"
    android:visibility="visible" />
//naštetih je le nekaj atributov, ampak dodeliti jih je mogoče še veliko več
```

Slika 6: Primer opisa elementa v XML

2. IZDELAVA APLIKACIJE

2.1. OSNOVNA IDEJA IN ZASNOVA IDEJE

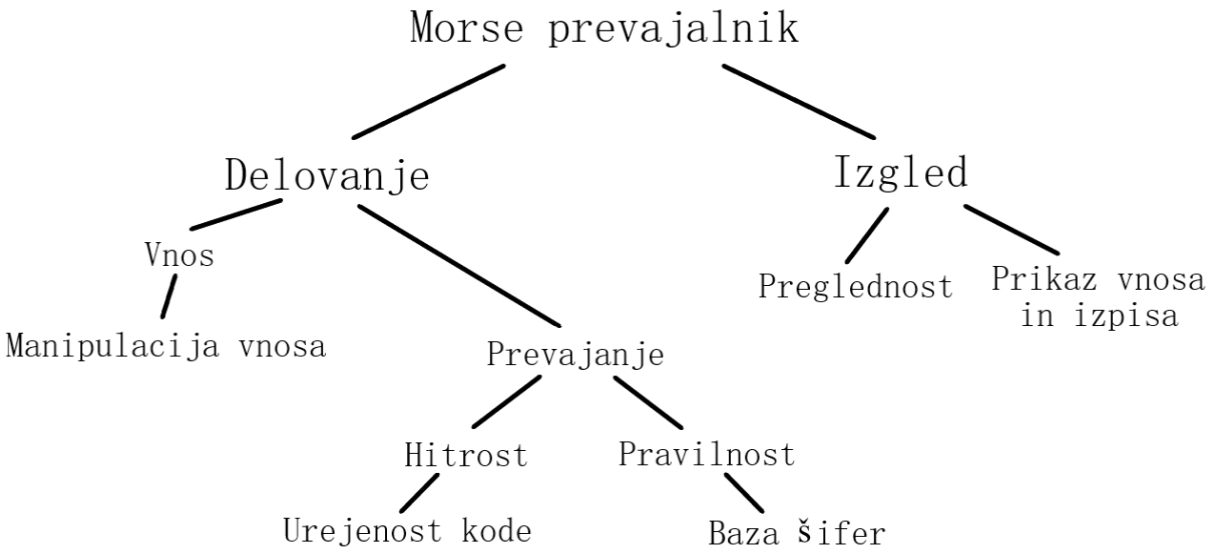
Za izdelavo aplikacije sem se odločil, ker sem v našem taborniškem društvu opazil potrebo po preprostem in dostopnem učilu za telegrafijo, ki je za otroke ena izmed zahtevnejših znanj, obravnavanih v taborništvu. Velika večina otrok, ki so včlanjeni in so dovolj stari za učenje signalizacije, imajo pametne telefone. Pri poučevanju sem želel imeti pripomoček, ki nazorno pokaže delovanje signalizacije in pomaga pri razumevanju, hkrati pa jim je dostopen ob vsakem trenutku. Shemo ciljev aplikacije vidimo na Sliki 7.



Slika 7: Shema ciljev aplikacije

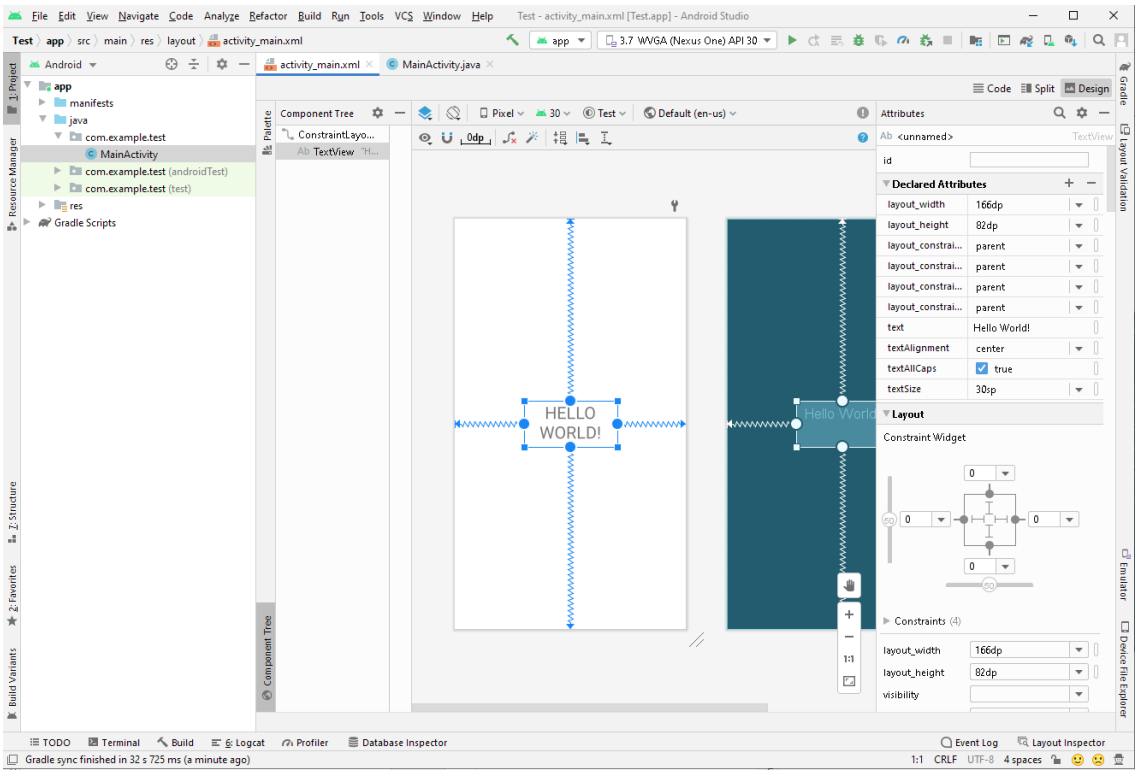
2.2. NAČRTOVANJE IZDELAVE

Prvi korak v procesu načrtovanja je pridobivanje literature in ostalih virov. Ker sem se z Javo spoznal že pri pouku računalništva in se telegrafijo naučil pri tabornikih, sem se v večini odločil za iskanje vsebine o Android Studiu, XML in aplikacijah. Na podlagi razumevanja principa delovanja šifriranja besedila v Morse kodo sem naredil funkcionalno dekompozicijo delovanja aplikacije, ki jo vidimo na Sliki 8. Upošteval sem tudi dejstvo, da je potrebno rezultate programa tudi pravilno prikazati in prilagoditi izgled za dobro uporabniško izkušnjo. Ključni del aplikacije je njena zmožnost prevajanja in manipulacije končnega rezultata, najbolj pomembno pa je kako pravilno, hitro in efektivno se prevajanje med Morsejevo abecedo in besedilom sploh izvede.



Slika 8: Funkcionalna dekompozicija aplikacije

S pripravljeno idejo in osnovnim konceptom končnega izdelka se je treba seznaniti s programskim okoljem, v katerem se piše aplikacijo. Android Studio ima znotraj urejevalnika pomoč pri spoznavanju programa in daje nasvete pri delu. Večinoma sem si pomagal z uradno dokumentacijo Android Studio, kjer piše vse o zgradbi in delovanju vseh knjižnic ter metod znotraj urejevalnika. Spoznavanje z Android Studio se je začelo s preprostimi preizkusi XML povezovanja z Javo v aktivnosti `MainActivity.java` in preizkušanjem dela z atributi, kakor vidimo na desni strani Slike 9. Spoznal sem se tudi z nekaj vmesniškimi elementi. Na Sliki 3 je razvidno delo z elementom prikaza besedila (angl. TextView) ter njegovo manipulacijo preko atributov za pozicijo, velikost besedila, oddaljenostjo od robov enkrana itd.



Slika 9: Izgled okolja Android Studio

2.3. PROGRAMIRANJE

2.3.1. JAVA

Prvi del sestavlja *MainActivity.java*, kjer je zapisan glavni del programa aplikacije in tvori hrbtenico aplikacije. Celotna datoteka je v Prilogi 4. Deklaracijo razreda *MainActivity* vidimo na Sliki 10.

```
public class MainActivity extends AppCompatActivity { ... }
```

Slika 10: Razred »MainActivity«

Tekom programiranja sem dodal vedno več različnih elementov, kar pomeni, da je bilo potrebno uporabiti veliko knjižnic. Med njimi so knjižnice za namen (angl. Intent), ki bodo pomagale s prikaznimi okni ter različni gumbi in prikazi. Vse knjižnice so zapisane pred deklaracijo razreda *MainActivity*, kot vidimo na Sliki 11.

```
import android.content.ClipData;
import android.content.ClipboardManager;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
```

Slika 11: Knjižnice v »MainActivity«

Za vsak element posebej je potrebna inicializacija, kot jo vidimo na Sliki 12 in končna deklaracija, kot jo vidimo na Sliki 13. Tabele, kjer so shranjene šifre znakov v Morsejevi abecedi sem zapisal tako, da so poravnane glede na indekse za lažje prevajanje z metodami. Gumbe, poglede in tabele sem inicializiral ter zapisal posebej za boljšo preglednost.

```
String[] morseCrke = {".-", "-...", "-.-.", "-..", ".", "..-.", "--.",
"....", "..", ".---", "-.-", ".-..", "--", "-.", "---", ".--.", ".-.",
"...", "-", "..-", "...-", "--..", "-..-", "-.--", "--.-", ".--", "//", "--
---", ".----", "..---", "...--", "....-", ".....", "-....", "--...", "---
..", "----.", ".-.-.-", "..--..", "--..--", "-.-.--", "-....-", "-.--.", "-
.--.-", "---...", "-.-.-.", "-...-", ".-.-.", "..--.-", ".--.-.", "-..-."};
String[] tekstCrke = {"a", "b", "c", "d", "e", "f", "g", "h", "i", "j",
"k", "l", "m", "n", "o", "p", "r", "s", "t", "u", "v", "z", "x", "y", "q",
"w", " ", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", ".", "?", ",",
"!", "-", "(", ")", ":", ";", "=", "+", "_", "@", "/"};
EditText Input;
TextView prevodText;
ImageButton gumb_prevodM;
ImageButton gumb_prevodT;
Button gumb_kop;
Button gumb_br;
Button gumb_menjaj;
Button gumb_helpMain;
ImageButton gumb_help;
```

Slika 12: Inicializacija spremenljivk v »MainActivity«

Za inicializacijo sem vse elemente deklariral iz povezanih XML datotek. Vsak ima svojo posebno ID vrednost, s katero lahko element navezujem na Java kodo. Končna deklaracija in zapis metod je potekal znotraj metode *onCreate()*, ki se v *MainActivity* navezuje na XML postavitev *activity_main*, le-to pa se nastavi z metodo *setContentView()*. (9)

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    setContentView(R.layout.activity_main);

Input = findViewById(R.id.theMainText);
prevodText = findViewById(R.id.textView2);
gumb_prevodM = findViewById(R.id.buttonM);
gumb_prevodT = findViewById(R.id.buttonT);
gumb_kop = findViewById(R.id.buttonkopiraj);
gumb_br = findViewById(R.id.buttonbrisi);
gumb_menjaj = findViewById(R.id.button_menjaj);
gumb_help = findViewById(R.id.button_help);
gumb_helpMain = findViewById(R.id.button_helpMain);

...
```

Slika 13: Deklaracija spremenljivk v »MainActivity«

Znotraj *onCreate()* so zapisane tudi vse metode, ki se navezujejo na zgoraj navedene elemente. Za vsak gumb posebej sem izvedel metodo *setOnClickListener()*, ki ima znotraj gnezdene ostale metode, kot so metode za nastavljanje besedila, uporabe metode za prevod, določitev vrednosti izpisa, pogoje ob klikih itd.

Gumb *gumb_menjaj* (Menjaj, angl. verzija Switch) sem ustvaril za preprosto menjavo besedila v polju vnosa in izpisa, kot vidimo na Sliki 14. Primer uporabe bi bil, da želimo prevedeno besedilo preoblikovati in prevesti še enkrat, ne želimo pa prepisovati izpisa. Ob kliku se besedili zamenjata. Vrednosti spremenljivk tipa String *zg_vr* in *sp_vr* sta pridobljeni z metodo *getText()*, da se lahko pozneje z metodo *setText()* vrednosti obrne. Koda za *gumb_menjaj*:

```
//gumb Menjaj
gumb_menjaj.setOnClickListener(v -> {
    String zg_vr = Input.getText().toString();
    String sp_vr = prevodText.getText().toString();

    prevodText.setText(zg_vr);
    Input.setText(sp_vr);
});
```

Slika 14: Gumb »Menjaj« (angl. verzija Switch)

Gumb *gumb_kop* (Kopiraj, angl. verzija Copy) sem ustvaril za kopiranje prevedenega besedila na odložišče. Ta funkcija je uporabna, če preko mobilne naprave urejamo neko besedilo, kjer želimo hitro dodati prevod. Lahko bi služilo tudi pri šifriranem sporočanju med osebama. Če imata obe osebi naložen Morse prevajalnik, si lahko dopisujeta s šifriranjem. V kodi sem deklariral String spremenljivko *vred*, ki z metodo *getText()* pridobi besedilo in se ga lahko uporabi naprej, kar vidimo na Sliki 15. Objekta *ClipboardManager* in *ClipData* sta vgrajena in služita pri delu z vsebino odložišča. *Toast* je poseben objekt, ki na zaslon izpiše začasno opozorilo npr. »Prosim vnesite besedilo!«. Koda za *gumb_kop*:

```
//gumb Kopiraj
gumb_kop.setOnClickListener(v -> {
    String vred = prevodText.getText().toString();
    String prosnja = getResources().getString(R.string.prosnja);
    String vsebina = getResources().getString(R.string.vsebina);
    String kopirano = getResources().getString(R.string.kopirano);

    if (vred.isEmpty()) {
        Toast.makeText(getApplicationContext(), prosnja,
Toast.LENGTH_SHORT).show();
    } else {
        ClipboardManager clipboardManager = (ClipboardManager)
getSystemService(Context.CLIPBOARD_SERVICE);
        ClipData clipData = ClipData.newPlainText(vsebina, vred);
        clipboardManager.setPrimaryClip(clipData);
        Toast.makeText(getApplicationContext(), kopirano,
Toast.LENGTH_SHORT).show();

    }
});
```

Slika 15: Gumb »Kopiraj« (angl. verzija Copy)

Gumb *gumb_br* sem ustvaril za brisanje vnosa in izpisa na prazna polja. Funkcija pride prav, ko želimo takoj pobrisati vse s polj za novo prevajanje. V kodi sem deklariral String spremenljivko *bes*, ki z metodo *getText()* pridobi besedilo iz vnosa, da lahko program zažene pogojni stavek s pogojem *bes.isEmpty()*. Kodo gumba za brisanje vidimo na Sliki 16. Če je vnos prazen se na zaslonu preko objekta *Toast* izpiše besedilo, da je vnos prazen, če pa ni, pa pobriše vnosno in izpisno polje. Koda za *gumb_br*:

```
//gumb Brisi
gumb_br.setOnClickListener(v -> {
    String bes = Input.getText().toString();
    String prazno = getResources().getString(R.string.prazno);
    if(bes.isEmpty()){
        Toast.makeText(getApplicationContext(),
prazno,Toast.LENGTH_SHORT).show();
    } else {
        Input.setText("");
        prevodText.setText("");
    }
});
```

Slika 16: Gumb za brisanje

Zadnja gumba sta *gumb_help* in *gumb_helpMain*, ki sem ju ustvaril za prikaz navodil in pomožne tabele Morsejeve abecede. Z njima si uporabnik pomaga pri uporabi aplikacije. V navodilih je zapisano besedilo z napotki za zapisovanje Morsejeve abecede za prevod, v tabeli z Morsejevo abecedo pa je našteta celotna abeceda z nekaj ločili in števili. Obema gumboma se z metodo *setOnClickListener()* pomaga za dostop do lastnih razredov. Gumb *gumb_help* ima glavno kodo v *popup.java*, *gumb_helpMain* pa v *popup2.java*. V gnezdeni metodi je potrebno izvesti metodo *startActivity()*, ki ustvari nov namen (angl. Intent) za dostop do razredov, kot vidimo na Sliki 17. (10) Koda za *gumb_help* in *gumb_helpMain*:

```
//prvi gumb za pomoč
    gumb_help.setOnClickListener(v -> startActivity(new Intent(MainActivity.this,
popup.class)));

//drugi gumb za pomoč
    gumb_helpMain.setOnClickListener(v -> startActivity(new
Intent(MainActivity.this, popup2.class)));
```

Slika 17: Gumba za pomoč

Drugi del sestavljata *popup.java* in *popup2.java*. Ustvaril sem ju za prikazovanje pojavnih oken z navodili in pomožno tabelo Morsejeve abecede. Obliki sta zelo podobni, ker se kliče pri obeh le drugo XML datoteko (*popup_window.xml* in *popup2_window.xml*), dostop pa je še vedno z glavne strani *activity_main*. Oba razreda sta odgovorna za prikazovanje pojavnih oken preko *setContentView()*, ki nastavi prikaz okna. Ker ne želim, da zapolni celoten zaslon, sem se odločil prilagoditi velikost okna. Okno je mogoče prilagoditi z *getWindowManager().getDefaultDisplay().getMetrics(dm)*, ki definira karakteristike okna oz. zaslona, preko metode *setLayout()* pa je mogoče prilagoditi okno na želeno velikost. (10) Kodo za *popup.java* in *popup2.java* vidimo na Sliki 18 in Sliki 19 oz. v Prilogi 5 in Prilogi 6.

```
package com.example.morseprevajalnik1;
import android.os.Bundle;
import android.util.DisplayMetrics;

public class popup extends MainActivity{

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.popup_window);

        //dolocanje prikaznih lastnosti prikaznega okna za pomoč 1
        DisplayMetrics dm = new DisplayMetrics();
        getWindowManager().getDefaultDisplay().getMetrics(dm);

        int visina = dm.heightPixels;
        int sirina = dm.widthPixels;

        getWindow().setLayout((int)(sirina *0.8),(int)(visina *0.5));
    }
}
```

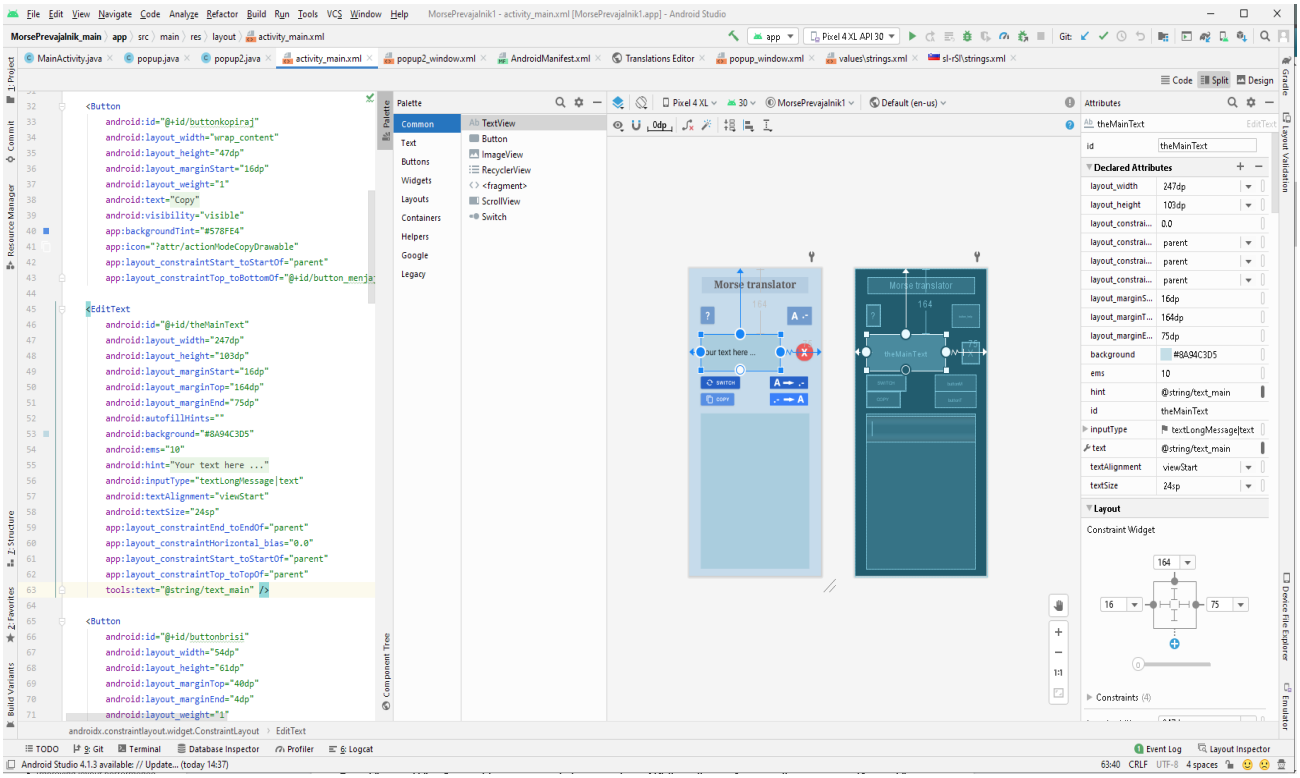
Slika 18: Razred »popup«

```java
package com.example.morseprevajalnik1;
import android.os.Bundle;
import android.util.DisplayMetrics;

public class popup2 extends MainActivity{

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.popup2_window);

        //dolocanje prikaznih lastnosti prikaznega okna za pomoč 2
        DisplayMetrics dm = new DisplayMetrics();
        getWindowManager().getDefaultDisplay().getMetrics(dm);

        int visina = dm.heightPixels;
        int sirina = dm.widthPixels;

        getWindow().setLayout((int)(sirina *0.8),(int)(visina *0.5));
    }
}
```

Slika 19: Razred »popup2«

2.3.2. XML

Android Studio zagotavlja pregledno pisanje in urejanje v glavnem programskem jeziku (lahko Java ali Kotlin) ter XML kode. Za XML dodatno omogoča urejanje kode posredno preko urejevanega vmesnika, ki da možnost grafičnega oblikovanja na princip **»drag-and-drop«**. (9)



Slika 20: Urejevalniški vmesnik za XML

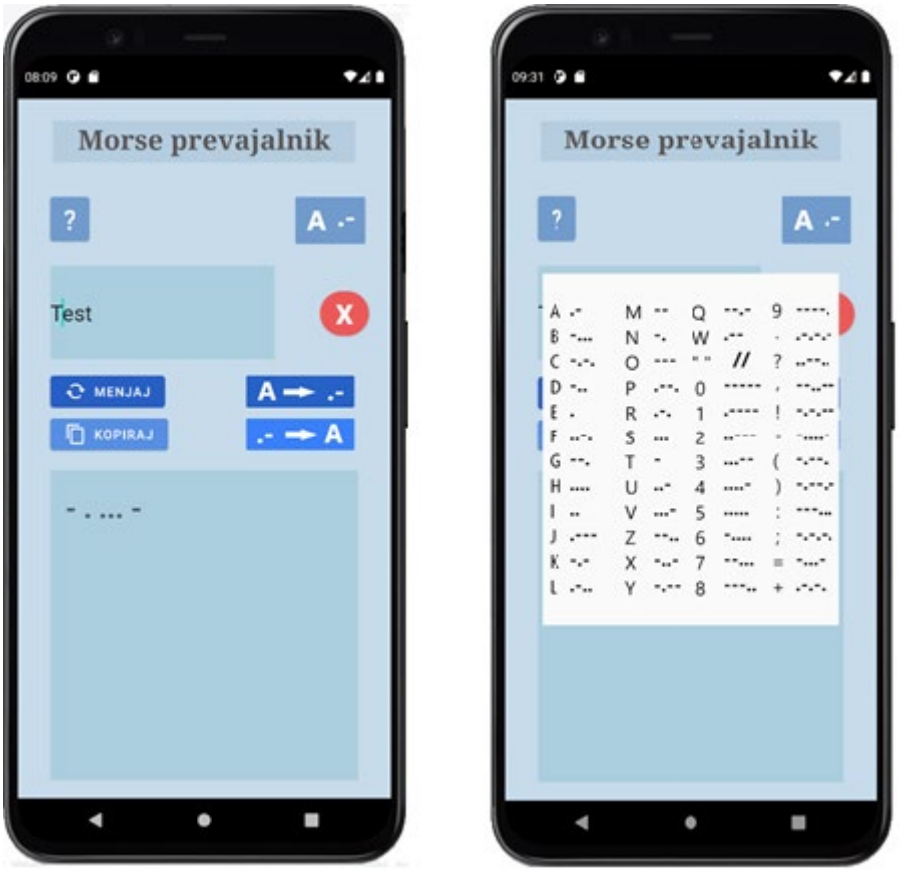
S pomočjo urejevalnika mi je bilo lažje sestaviti izgled aplikacije in ga prilagoditi za najbolj optimalno uporabniško izkušnjo. Z urejevalnikom sem elemente postavil v glavno pozicijo in nastavil mejnike oz. določil, kateri elementi so nadrejeni in kateri podrejeni (»parent-child« relacije). Vmesnik še posebej olajša delo z razdeljenim prikazom med kodo in grafičnim prikazom, kakor vidimo na Sliki 20. Da je iskanje in urejanje elementa hitro in enostavno, to dosežemo z iskanjem v grafičnem delu ter urejanjem v kodi na strani. Če smo pozorni je jasno tudi gnezdenje elementov glede na zaključeno notacijo XML kode (nakazano z »<« in »/>«). Za dodajanje in urejanje atributov pripomore tabela atributov na desni strani urejevalnika, kjer je za vsak element naveden vsak pripadajoči atribut.

Za pravilno in urejeno delovanje aplikacije je potrebno nekaj ključnih elementov. Glavni sta polji za vnos besedila in prikaz prevedenega besedila. Večina gumbov aplikacije je odgovorna za manipulacijo s tema poljema. Najpomembnejša gumba sta gumba za prevajanje iz besedila v Morsejevo abecedo in za prevajanje iz Morsejeve abecede v besedilo. Za nedvoumno uporabo imata gumba za prikaz sliko, ki nazorno pokaže funkcijo gumba. Poleg gumbov za prevod ima aplikacija še dva gumba za delo s poljema, ki omogočata menjavo besedil med poljema in kopiranje prevedenega besedila v odložišče in dva gumba za pomoč, ki izpišeta navodila in tabelo Morsejeve abecede. Vsi elementi aplikacije imajo svojo XML kodo, ki jo vidimo v Prilogi 1, Prilogi 2 in Prilogi 3.

Aplikacijo sem prilagodil za pokončni prikaz na mobilni napravi, ker je tako najbolj pregledna in posledično uporabna. Med uporabo je pomembno, da ima uporabnik dovolj pomoči vmesnika, ki s svojo postavitvijo in preglednostjo pospeši uporabnikovo delo znotraj aplikacije. Pokončna postavitev zato pripomore že pri tem, da je uporaba možna samo z eno roko.

Tudi postavitev gumbov je zelo pomembna. Ključno je bilo približati gumbe, ki imajo podobne funkcije in jih dati na mesto, primerno njihovim funkcijam. Glavni gumbi se nahajajo v sredini prikaza med poljema besedil. Gumba za prevajanje besedila in gumb za brisanje so na desni strani za hiter pritisk s palcem ob uporabi desne roke (privzeta za desničarje), na levi pa sta gumba za menjavo in kopiranje besedila. Manj ključna gumba za pomoč sta blizu vrha prikaza, da ne ovirata uporabnika, sta pa dovolj vidna, da ju uporabnik vidi in prepozna njuno funkcijo.

Največ težav sem srečal ravno pri delu z XML, ker je izgled aplikacije zelo tesno povezan z uporabniško izkušnjo, ki pa je edina stvar, za katero lahko rečemo, da loči navadne programe od mobilnih aplikacij. Uporabnik se najprej osredotoči na to, kar vidi pred sabo preden začne z uporabo. Izgled lahko na različne načine spremeni uporabo. Lahko jo pospeši, upočasni, poenostavi, oteži, onemogoči itd. Načrtovanje XML kode je potekalo najdlje časa in je bilo deležno največjega števila popravkov in prilagoditev, ker je Java koda dovolj jasna že od začetka in njeno delovanje ne vpliva na uporabnika, ker se dogaja v ozadju, XML pa strmi v nas odkar smo odprli aplikacijo. Težave sem rešil tako, da sem načrtoval medsebojne razdalje elementov ter razdalje med elementi in robovi zaslona. Podobno je bilo potrebno prilagoditi velikost pojavnega okna za pomoč, da ni zastrl celotnega zaslona. Končni izgled vidimo na Sliki 21.



Slika 21: Izgled aplikacije brez pomoči (levo) in s pomočjo tabele (desno)

Aplikaciji sem dodal vrednosti besedil in besed za slovenščino in angleščino. Od nastavitev jezika naprave je odvisno, v katerem jeziku bo aplikacija. To sem dosegel z ločenimi XML datotekami za slovenske in angleške String vrednosti, kot vidimo na Sliki 22 in Sliki 23. Android Studio omogoča vezavo datoteke z izbranim jezikom tako, da ko prepozna nastavitev v napravi, nastavi vrednost jezika tudi v aplikaciji. Angleško je privzeto za vse neslovenske jezike, ko pa imamo v napravi izbrano slovenščino, se lahko poslužujemo slovenske verzije. Besede, ki so lahko različne za angleščino so vsa pojavna besedila in napisi na gumbih ter poljih. S tem omogočim uporabo otrokom in tistim, ki ne govorijo angleško ter tudi obratno za tiste, ki ne govorijo slovensko.

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Morse translator</string>
    <string name="text_main">Your text here ...</string>
    <string name="buttonM">Encode</string>
    <string name="buttonT">Decode</string>
    <string name="buttonkopiraj">Copy</string>
    <string name="prosnja">Please insert text!</string>
    <string name="kopirano">Copied to Clipboard</string>
    <string name="prazno">Empty</string>
    <string name="menjaj">Switch</string>
    <string name="vsebina">Data</string>
    <string name="text_second"></string>
    <string name="nic" translatable="false"></string>
    <string name="x">X</string>
    <string name="pomoc">To write in Morse you must write in dots and
dashes ( . and - ). If you would like to finish one letter you hit
\"Space\" once and continue. To finish a word hit \"Space\", followed by
two slashes ( // ) and hit \"Space\" one more time. (Example: .- // -. . -)
Without hitting a \"Space\" the translation will be faulty.</string>
</resources>
```

Slika 22: Angleške String vrednosti

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
    <string name="app_name">Morse prevajalnik</string>
    <string name="text_main">Besedilo tukaj ...</string>
    <string name="buttonkopiraj">Kopiraj</string>
    <string name="prosnja">"Prosim, vnesite besedilo! "</string>
    <string name="vsebina">Vsebina</string>
    <string name="kopirano">Kopirano v odložišče</string>
    <string name="prazno">Prazno</string>
    <string name="menjaj">menjaj</string>
    <string name="buttonT">Dekodiraj</string>
    <string name="buttonM">Kodiraj</string>
    <string name="text_second" />
    <string name="x">X</string>
    <string name="pomoc">Za pisanje v Morseju morate pisati s pikami in
pomišljaji (. in -). Če želite dokončati eno črko, enkrat pritisnite
\"Presledek\" in nadaljujte. Za zaključek besede pritisnite \"Presledek\",
sledita dve poševnici (//) in še enkrat pritisnite \"Presledek\". Brez
pritiska na \"presledek\" bo prevod napačen.</string>
</resources>
```

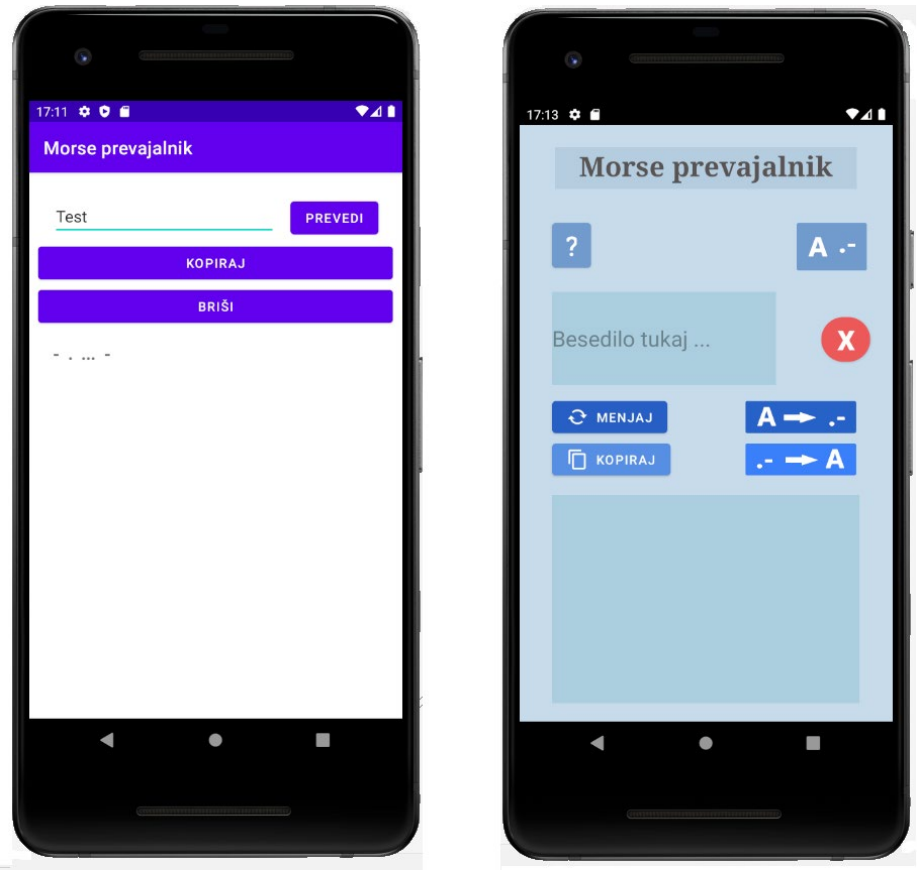
Slika 23: Slovenske String vrednosti

3. TESTIRANJE IN KONČNI REZULTAT

Proces testiranja je trajal od trenutka, ko sem napisal prvih nekaj vrstic v Javi. Sicer zelo primitivna, a delujoča verzija, je omogočala vnos besedila v polje in prevod v Morsejevo abecedo, kot vidimo na Sliki 24. Na tej točki sem se že dovolj dobro seznanil z urejevalnikom, da sem lahko suvereno dodajal, spreminjal in posodabljal svojo kodo. Kmalu po uspešnem prevajanju sem začel razmišljati o izgledu aplikacije in se posvetil pomembnosti prvega vtisa ob zagonu aplikacije. Zagotovil sem, da ima aplikacija nazorne gumbe z jasnimi funkcijami.

Samostojno testiranje sem nadomestil in pokazal aplikacijo družini, prijateljem in sošolcem. Vsi so aplikacijo preizkusili in mi ponudili vrsto opomb, kritik in vprašanj. Veliko jih je opomnilo na izgled in posebej so poudarili pomembnost navodil za uspešno uporabo. Nekateri so predlagali tudi možnost za dodajanje dodatnih funkcij, kot je na primer zvočna interpretacija prevedenega besedila, česar pa v končno verzijo aplikacije nisem uspel dodati.

Pri testiranju sem opazil, da so imeli uporabniki največ težav pri uporabi gumbov. V starejših različicah aplikacije niso bili gumbi označeni z ikonami in slikami, ampak le z besedami kot so »kodiraj« in »dekodiraj«, kar jih je v večini zmedlo. Rezultat te ugotovitve je lepši in preglednejši izgled gumbov ter jasnost njihovih funkcij. Podobno je bilo težavno delo pri prevajanju v različne smeri, za katero je bilo prej namenjeno stikalo, ki je spremenilo način prevajanja. To sem rešil z možnostjo prevajanja ob pritisku na dva različna gumba za različna načina prevajanja. Z nekaj dodatnimi popravki je bila aplikacija končana.



Slika 24: Izgled na začetku (levo) in končni izdelek (desno)

4. ZAKLJUČEK

Pisanje te seminarske naloge mi je razširilo obzorja in me postavilo pred preizkus lastnih sposobnosti in znanja. Dolgotrajno delo in trud sta na koncu pokazala, da sem sposoben pridobiti dobre in obetajoče rezultate. Končna aplikacija je odraz mojega dela in mi pove, da sem naredil velik korak v pravo smer za mojo prihodnost v razvijalskih vodah. Predvsem sem se veliko naučil o ustvarjanju aplikacij, spoznal sem se z okolji za pisanje aplikacij, preizkusil sem nove pristope k povezovanju programskih jezikov in drugih znanj. Končne cilje sem z izdelkom dosegel in izpolnil vse načrte, katere sem si postavil pri načrtovanju. Večina preizkusov je bilo uspešnih in pridobil sem koristne informacije za dodatno nadgradnjo. Naučil sem se pomembnosti uporabniške izkušnje in kako pomembno je biti pozoren na vse povratne informacije uporabnikov.

Aplikacija ima prostora še za dodatne izboljšave. Lahko bi ji dodal možnost pretvorbe v zvočne ali svetlobne signale, ki bi s pomočjo prevajalnika spremenili napravo v Morse oddajnik. V te izboljšave se nisem želel spuščati, da ne bi po nepotrebnem porabil preveč časa na tej funkciji kakor na drugih bolj ključnih delih aplikacije, je pa načrt za prihodnost, ker nameravam aplikacijo dodelati in mogoče izdati za javno uporabo.

5. ZAHVALA

Zahvaljujem se mentorju, ki mi je odgovoril na vsa vprašanja in svetoval pri pisanju seminarske naloge. Zahvaljujem se svojim prijateljem in sošolcem, ki so z mano delili mnenja, mi svetovali pri izboljšavah in z menoj delili znanje. Zahvaljujem se svoji družini za vse spodbudne besede, potrpežljivost in nasvete v času pisanja te naloge ter ker so mi stali ob strani. Na koncu se zahvaljujem tudi vsem ostalim, ki so kakorkoli pripomogli k mojemu delu.

6. VIRI IN LITERATURA

1. **Facka.** *Telegraf.* [Elektronski] [Navedeno: 20. marec 2021.] http://www.facka.si/gradiva/geo/amerika/8zanimivosti/telegraf.html.

2. **Jogan, Nejc, in drugi.** *V naravo: priročnik z nasveti za gibanje, bivanje in prehranjevanje v naravi.* Ljubljana : Zveza tabornikov Slovenije, nacionalna skavtska organizacija, 2012. str. 197-198. 978-961-6134-42-2.

3. **Android Developers.** Android Developers. *Meet Android Studio.* [Elektronski] [Navedeno: 15. marec 2021.] https://developer.android.com/studio/intro.

4. **Android Developers.** Android Developers. *View.OnClickListener.* [Elektronski] [Navedeno: 20. marec 2021.] https://developer.android.com/reference/android/view/View.OnClickListener.

5. **Android Developers.** Android Developers. *View.* [Elektronski] [Navedeno: 20. marec 2021.] https://developer.android.com/reference/android/view/View.

6. **Android Developers.** Android Developers. *TextView.* [Elektronski] [Navedeno: 20. marec 2021.] https://developer.android.com/reference/android/widget/TextView.

7. **Harold, Elliotte Rusty.** *XML Bible, Second Edition.* New York : Hungry Minds, Inc., 2001. 0-7645-4760-7.

8. **Dykes, Lucinda in Tittel, Ed.** *XML for Dummies, 4th Edition.* s.l. : John Wiley & Sons Inc., 2005. 9780764588457.

9. **Android Developers.** Android Developers. *Layouts.* [Elektronski] [Navedeno: 24. marec 2021.] https://developer.android.com/guide/topics/ui/declaring-layout.

10. **Vujović, Filip.** How To Create Pop Up Window In Android. [Elektronski] 8. maj 2015. [Navedeno: 14. marec 2021.] https://www.youtube.com/watch?v=fn5OlqQuOCk.

11. **Živković, Dejan.** *Osnove Java programiranja.* s.l. : Univerzitet Singidunum, 2018. 978-86-7912-558-3.

12. **Davison, Andrew.** *Killer Game Programming in Java.* s.l. : O'Reilly, 2005.

13. **Gerber, Adam in Clifton, Craig.** *Learn Android Studio: Build Android Apps Quickly and Effectively.* s.l. : Springer-Verlag Berlin and Heidelberg GmbH & Co. KG, 2015. 9781430266013.

14. **Gok, Nizamettin in Khanna, Nittin.** *Building Hybrid Android Apps with Java and JavaScript.* s.l. : O'Reilly, 2013.

15. **Juntao Yuan, Michael.** *Enterprise J2ME: Developing Mobile Java Applications.* s.l. : Prentice Hall PTR, 2004.

16. **Klawonn, Frank.** *Introduction to Computer Graphics: Using Java 2D and 3D.* s.l. : Springer, 2008.

17. **Mesojedec, Uroš in Fabjan, Borut.** *Java 2: temelji programiranja.* s.l. : Založba Pasadena, 2004. 9616361309.

18. **Mesojedec, Uroš.** *Java, programiranje za internet.* s.l. : Založba Pasadena, 1997. 9616065289.

19. **Stephen Horstmann, Cay in Cornell, Gary.** *Core Java 2: Volume I Fundamentals.* s.l. : Prentice Hall PTR, 2000.

20. **Živković, Dejan.** *Java programiranje.* s.l. : Univerzitet Singidunum, 2019. 978-86-7912-521-7.

7. STVARNO KAZALO

8. PRILOGE

Priloga 1: Activity_main.xml

```
<?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="#ABABC9E1"
    android:importantForAccessibility="yes"
    android:padding="20dp"
    android:textAlignment="center"
    tools:context=".MainActivity">

    <ImageButton
        android:id="@+id/buttonM"
        android:layout_width="wrap_content"
        android:layout_height="47dp"
        android:layout_marginTop="12dp"
        android:layout_marginEnd="16dp"
        android:adjustViewBounds="true"
        android:contentDescription="@string/nic"
        android:scaleType="fitCenter"
        android:scrollbars="vertical"
        android:src="@drawable/av__"
        android:text="@string/buttonM"
        android:visibility="visible"
        app:backgroundTint="#2862C8"
        app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintTop_toBottomOf="@+id/theM
ainText" />

    <Button
        android:id="@+id/buttonkopiraj"
        android:layout_width="wrap_content"
        android:layout_height="47dp"
        android:layout_marginStart="16dp"
        android:layout_weight="1"
        android:text="@string/buttonkopiraj"
        android:visibility="visible"
        app:backgroundTint="#578FE4"
        app:icon="?attr/actionModeCopyDrawable"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/butto
n_menjaj" />

    <EditText
        android:id="@+id/theMainText"
        android:layout_width="247dp"
        android:layout_height="103dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="164dp"
        android:layout_marginEnd="75dp"
        android:autofillHints=""
        android:background="#8A94C3D5"
        android:ems="10"
        android:hint="@string/text_main"
        android:inputType="textLongMessage|text"
        android:textAlignment="viewStart"
        android:textSize="24sp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.0"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent"
        tools:text="@string/text_main" />

    <Button
        android:id="@+id/buttonbrisi"
        android:layout_width="54dp"
        android:layout_height="61dp"
        android:layout_marginTop="40dp"
        android:layout_marginEnd="4dp"
        android:layout_weight="1"
        android:focusable="false"
        android:fontFamily="sans-serif-black"
        android:text="@string/x"
        android:textAlignment="center"
        android:textSize="30sp"
        android:textStyle="bold"
        app:backgroundTint="#ED5959"
        app:cornerRadius="60dp"
        app:iconPadding="0dp"
        app:iconTint="@color/white"
        app:iconTintMode="src_atop"
        app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintTop_toBottomOf="@+id/butto
n_help" />

    <ScrollView
        android:id="@+id/scrollMain"
        android:layout_width="0dp"
        android:layout_height="0dp"
        android:layout_marginStart="16dp"
        android:layout_marginTop="16dp"
        android:layout_marginEnd="16dp"
        android:background="#8A94C3D5"
        android:isScrollContainer="true"

android:scrollbarAlwaysDrawVerticalTrack="true"

app:layout_constraintBottom_toBottomOf="parent"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintHorizontal_bias="0.5"
        app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/butto
nkopiraj">
```

```
<LinearLayout
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginTop="5dp"
        android:orientation="vertical">


        <TextView
            android:id="@+id/textView2"
            android:layout_width="wrap_content"
            android:layout_height="wrap_content"
            android:layout_margin="16dp"
            android:layout_marginStart="16dp"
            android:layout_marginLeft="6dp"
            android:layout_marginTop="16dp"
            android:layout_marginRight="6dp"
            android:layout_marginBottom="16dp"
            android:fontFamily="sans-serif-black"
            android:textSize="30sp"
            android:textStyle="bold"
            tools:layout_margin="16dp"
            tools:text="@string/text_second" />
    </LinearLayout>
  </ScrollView>
  <Button
    android:id="@+id/button_menjaj"
    android:layout_width="wrap_content"
    android:layout_height="47dp"
    android:layout_marginStart="16dp"
    android:layout_marginTop="12dp"
    android:focusable="false"
    android:text="@string/menjaj"
    app:backgroundTint="#255FC5"

app:icon="@android:drawable/stat_notify_sync"
    app:layout_constraintStart_toStartOf="parent"

app:layout_constraintTop_toBottomOf="@+id/theM
ainText"
    tools:text="@string/menjaj" />

  <ImageButton
    android:id="@+id/buttonT"
    android:layout_width="wrap_content"
    android:layout_height="47dp"
    android:layout_marginEnd="16dp"
    android:adjustViewBounds="true"
    android:contentDescription="@string/nic"
    android:scaleType="fitCenter"
    android:src="@drawable/__va"
    android:text="@string/buttonT"
    app:backgroundTint="#3A80FA"
    app:layout_constraintEnd_toEndOf="parent"
app:layout_constraintTop_toBottomOf="@+id/butto
nM" />
  <!--suppress AndroidDomInspection -->
  <ImageButton
    android:id="@+id/button_help"
    android:layout_width="85dp"
    android:layout_height="64dp"
    android:layout_marginTop="32dp"
    android:layout_marginEnd="4dp"
    android:adjustViewBounds="true"
    android:contentDescription="@string/nic"
    android:cropToPadding="true"
    android:scaleType="fitCenter"
    android:textAlignment="center"
    android:textAllCaps="true"
    android:textColor="#FFFFFF"
    android:textSize="24sp"
    android:textStyle="bold"
    app:backgroundTint="#729BCD"
    app:cornerRadius="43dp"
    app:layout_constraintEnd_toEndOf="parent"

app:layout_constraintTop_toBottomOf="@+id/text
View"
    app:srcCompat="@drawable/znakigumb" />

  <TextView
    android:id="@+id/textView"
    android:layout_width="333dp"
    android:layout_height="46dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="4dp"
    android:layout_marginEnd="8dp"
    android:background="#5C97B5CD"
    android:fontFamily="serif"
    android:text="@string/app_name"
    android:textAlignment="center"

android:textAppearance="@style/TextAppearance.Ap
pCompat.Display1"
    android:textColor="#595454"
    android:textSize="30sp"
    android:textStyle="bold"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />

</androidx.constraintlayout.widget.ConstraintLayou
t>
```

Priloga 2: Popup_window.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout 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="#00FFFFFF">

    <ImageView
        android:id="@+id/imageView"
        android:layout_width="412dp"
        android:layout_height="match_parent"
        android:layout_alignParentStart="true"
        android:layout_alignParentEnd="true"
        android:layout_centerHorizontal="true"
        android:layout_marginStart="0dp"
        android:layout_marginEnd="-1dp"
        app:srcCompat="@drawable/morseslika1" />
</RelativeLayout>
```

Priloga 3: Popup2_window.xml

```
<?xml version="1.0" encoding="utf-8"?>
<RelativeLayout xmlns:android="http://schemas.android.com/apk/res/android"
          android:layout_width="match_parent" android:layout_height="match_parent">

    <TextView
        android:id="@+id/help_text"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentStart="true"
        android:layout_alignParentTop="true"
        android:layout_alignParentEnd="true"
        android:layout_centerInParent="true"
        android:layout_marginStart="12dp"
        android:layout_marginTop="62dp"
        android:layout_marginEnd="11dp"
        android:text="@string/pomoc"
        android:textAlignment="gravity"
        android:textSize="18sp" />
</RelativeLayout>
```

Priloga 4: MainActivity.java

```
package com.example.morseprevajalnik1;

import android.content.ClipData;
import android.content.ClipboardManager;
import android.content.Context;
import android.content.Intent;
import android.os.Bundle;
import android.widget.Button;
import android.widget.EditText;
import android.widget.ImageButton;
import android.widget.TextView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {

    String[] morseCrke = {".-", "-...", "-.-.", "-..", ".", "..-.", "--.", "....", "..", ".---", "-.-", ".-..", "--", "-.", "---", ".--.", ".-.", "...", "-", "..-", "...-", "--..", "-..-", "-.--", "--.-", ".--", "//", "-----", ".----", "..---", "...--", "....-", ".....", "-....", "--...", "---..", "----.", ".-.-.-", "..--..", "--..--", "-.-.--", "-....-", "-.--.", "-.--.-", "---...", "-.-.-.", "-...-", ".-.-.", "..--.-", ".--.-.", "-..-."};
    String[] tekstCrke = {"a", "b", "c", "d", "e", "f", "g", "h", "i", "j", "k", "l", "m", "n", "o", "p", "r", "s", "t", "u", "v", "z", "x", "y", "q", "w", " ", "0", "1", "2", "3", "4", "5", "6", "7", "8", "9", ".", "?", ",", "!", "-", "(", ")", ":", ";", "=", "+", "_", "@", "/"};

    EditText Input;
    TextView prevodText;
    ImageButton gumb_prevodM;
    ImageButton gumb_prevodT;
    Button gumb_kop;
    Button gumb_br;
    Button gumb_menjaj;
    Button gumb_helpMain;
    ImageButton gumb_help;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);

        Input = findViewById(R.id.theMainText);
        prevodText = findViewById(R.id.textView2);
        gumb_prevodM = findViewById(R.id.buttonM);
        gumb_prevodT = findViewById(R.id.buttonT);
        gumb_kop = findViewById(R.id.buttonkopiraj);
        gumb_br = findViewById(R.id.buttonbrisi);
        gumb_menjaj = findViewById(R.id.button_menjaj);
        gumb_help = findViewById(R.id.button_help);
        gumb_helpMain = findViewById(R.id.button_helpMain);


        //gumb Menjaj
        gumb_menjaj.setOnClickListener(v -> {
            String zg_vr = Input.getText().toString();
            String sp_vr = prevodText.getText().toString();
            prevodText.setText(zg_vr);
            Input.setText(sp_vr);
        });
```

```
        //gumba Kodiraj in Dekodiraj
          //po kliku na gumb 'Kodiraj' izvede MorsePrevod
        gumb_prevodM.setOnClickListener(view -> {
            prevodText.setText(MorsePrevod(Input.getText().toString()));
        });

          //po kliku na gumb 'Dekodiraj' izvede TextPrevod
        gumb_prevodT.setOnClickListener(view -> {
            prevodText.setText(TextPrevod(Input.getText().toString()));
        });

        //gumb Kopiraj
        gumb_kop.setOnClickListener(v -> {
            String vred = prevodText.getText().toString();
            String prosnja = getResources().getString(R.string.prosnja);
            String vsebina = getResources().getString(R.string.vsebina);
            String kopirano = getResources().getString(R.string.kopirano);
            if (vred.isEmpty()) {
                        Toast.makeText(getApplicationContext(), prosnja, Toast.LENGTH_SHORT).show();
            } else {
                ClipboardManager clipboardManager = (ClipboardManager)
getSystemService(Context.CLIPBOARD_SERVICE);
                ClipData clipData = ClipData.newPlainText(vsebina, vred);
                clipboardManager.setPrimaryClip(clipData);
                Toast.makeText(getApplicationContext(), kopirano, Toast.LENGTH_SHORT).show();
            }});
        //gumb Brisi
        gumb_br.setOnClickListener(v -> {
            String bes = Input.getText().toString();
            String prazno = getResources().getString(R.string.prazno);
            if(bes.isEmpty()){
                Toast.makeText(getApplicationContext(), prazno,Toast.LENGTH_SHORT).show();
            } else {
                Input.setText("");
                prevodText.setText("");
            }
        });

        //gumb za pomoč
        gumb_help.setOnClickListener(v -> startActivity(new Intent(MainActivity.this, popup.class)));

        //gumb za pomoč
        gumb_helpMain.setOnClickListener(v -> startActivity(new Intent(MainActivity.this, popup2.class)));
    }

    //prevajanje v besedilo
    private String TextPrevod(String str) {
        String textSifrirano = "";
        String[] crke = str.split(" ");

        for (String crka : crke) {
            for (int j = 0; j < morseCrke.length; j++) {
                if (crka.equalsIgnoreCase(morseCrke[j])) {
                    textSifrirano = textSifrirano.concat(tekstCrke[j]);
                }
            }
        }
      return textSifrirano;
    }
```

```
    //prevajanje v Morse
    private String MorsePrevod(String str) {
        String morseSifrirano = "";
        String[] crke = str.split("");

        for (String crka : crke) {
            for (int j = 0; j < tekstCrke.length; j++) {
                if (crka.equalsIgnoreCase(tekstCrke[j])) {
                    morseSifrirano = morseSifrirano.concat(morseCrke[j]).concat(" ");
                }
                else if(crka.equals("č")){
                    morseSifrirano = morseSifrirano.concat("-.-.").concat(" ");
                    break;
                } else if(crka.equals("š")){
                    morseSifrirano = morseSifrirano.concat("...").concat(" ");
                    break;
                } else if(crka.equals("ž")){
                    morseSifrirano = morseSifrirano.concat("--..").concat(" ");
                    break;
                }
            }
        }
        return morseSifrirano;
    }
}
```

Priloga 5: Popup.java

```
package com.example.morseprevajalnik1;
import android.os.Bundle;
import android.util.DisplayMetrics;

public class popup extends MainActivity{

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);

        setContentView(R.layout.popup_window);

        //dolocanje prikaznih lastnosti prikaznega okna za pomoč
        DisplayMetrics dm = new DisplayMetrics();
        getWindowManager().getDefaultDisplay().getMetrics(dm);
        int visina = dm.heightPixels;
        int sirina = dm.widthPixels;
        getWindow().setLayout((int)(sirina *0.8),(int)(visina *0.45));
    }
}
```

Priloga 6: Popup2.java

```
package com.example.morseprevajalnik1;
import android.os.Bundle;
import android.util.DisplayMetrics;

public class popup2 extends MainActivity{

   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      setContentView(R.layout.popup2_window);

      //dolocanje prikaznih lastnosti prikaznega okna za pomoč
      DisplayMetrics dm = new DisplayMetrics();
      getWindowManager().getDefaultDisplay().getMetrics(dm);
      int visina = dm.heightPixels;
      int sirina = dm.widthPixels;
      getWindow().setLayout((int)(sirina *0.8),(int)(visina *0.45));
   }
}
```

Priloga 7: Strings.xml

```
<resources>
   <string name="app_name">Morse translator</string>
   <string name="text_main">Your text here ...</string>
   <string name="buttonM">Encode</string>
   <string name="buttonT">Decode</string>
   <string name="buttonkopiraj">Copy</string>
   <string name="prosnja">Please insert text!</string>
   <string name="kopirano">Copied to Clipboard</string>
   <string name="prazno">Empty</string>
   <string name="menjaj">Switch</string>
   <string name="vsebina">Data</string>
   <string name="text_second"></string>
   <string name="nic" translatable="false"></string>
   <string name="x">X</string>
   <string name="pomoc">To write in Morse you must write in dots and dashes ( . and - ). If you would like to finish one letter you hit \"Space\" once and continue. To finish a word hit \"Space\", followed by two slashes ( // ) and hit \"Space\" one more time. (Example: .- // -. . -) Without hitting a \"Space\" the translation will be faulty.</string>
</resources>
```

Priloga 8: Strings(SI).xml

```
<?xml version="1.0" encoding="utf-8"?>
<resources>
   <string name="app_name">Morse prevajalnik</string>
   <string name="text_main">Besedilo tukaj ...</string>
   <string name="buttonkopiraj">Kopiraj</string>
   <string name="prosnja">"Prosim, vnesite besedilo! "</string>
   <string name="vsebina">Vsebina</string>
   <string name="kopirano">Kopirano v odložišče</string>
   <string name="prazno">Prazno</string>
   <string name="menjaj">menjaj</string>
   <string name="buttonT">Dekodiraj</string>
   <string name="buttonM">Kodiraj</string>
   <string name="text_second" />
   <string name="x">X</string>
   <string name="pomoc">Za pisanje v Morseju morate pisati s pikami in pomišljaji (. in -). Če želite dokončati eno črko, enkrat pritisnite \"Presledek\" in nadaljujte. Za zaključek besede pritisnite \"Presledek\", sledita dve poševnici (//) in še enkrat pritisnite \"Presledek\". Brez pritiska na \"presledek\" bo prevod napačen.</string>
</resources>
```

Priloga 9: USB ključ s projektnimi datotekami