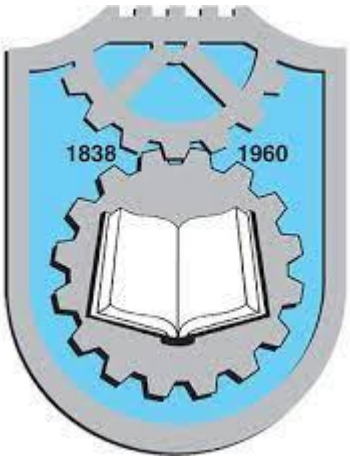
Fakultet inženjerskih nauka
Univerzitet u Kragujevcu



Programiranje mobilnih aplikacija

Projektni zadatak:
Aplikacija za puštanje muzike (eng. *Music player*)

Student:
Đorđe Molnar 660/2019

Predmetni nastavnik:
Vukašin Slavković

Sadržaj:

Opis korišćenog razvojnog okruženja:

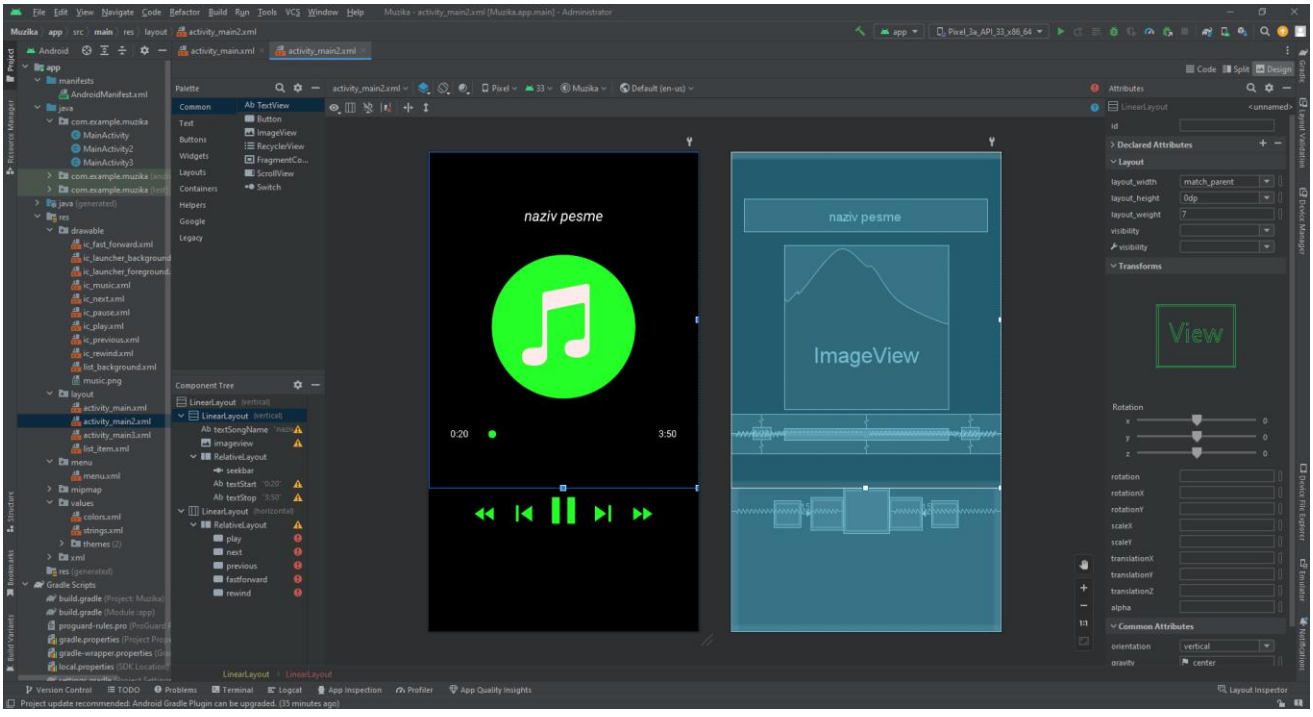
Android studio:

Za razvoj Android aplikacija koristi se alat Android Studio, koji je moguće instalirati na Windows, Linux i MacOS platformama. Android Studio je integrisano razvojno okruženje koje je bazirano na Intellij razvojnom okruženju.

Android Studio je razvojno okruženje zasnovano na IntelliJ IDEA softveru. Namenjen prvenstveno za razvoj Android mobilnih aplikacija, Android Studio je optimalno okruženje koje pruža sve mogućnosti trenutno dostupne u ovoj grani programiranja. Android aplikacije se mogu pisati i u drugim razvojnim okruženjima (na primer Eclipse IDE ili Visual Studio), ali sam razvoj je znatno otežan, jer druga okruženja nisu namenjena isključivo za Android i zahtevaju dodatna podešavanja u vidu dodataka specifičnih za Android projekte.

Budući da je Android Studio zasnovan na IntelliJ editoru koda i razvojnim alatima istog, sasvim je opravdano reći da je okruženje koje omogućava kompletno iskustvo u razvoju Android projekata.

Korisnički interfejs je kreiran tako da su najkorišćenije funkcionalnosti uvek dostupne na vidljivim mestima, a takođe i pruža mogućnost reorganizacije paleta alata onako kako korisniku odgovara. Pored ove dve karakteristike, Android Studio pruža i više nego dovoljno opcija za neometani razvoj Android aplikacija, uključujući podešavanja različitih perspektiva za prikaz strukture projekta, integraciju sa nekoliko različitih alata za kontrolu koda (engl. *version control/source control*) i mnoge druge.



Opis korišćenih tehnologija:

Java:

Java je objektno-orijentisani programski jezik, koji je razvila kompanija Sun Microsystems početkom 1990-ih godina.

Mnogi koncepti Jave su zasnovani na jeziku Oberon, Niklausa Virta, tvorca Paskala, Module i drugih jezika, i Hanspetera Mesenbeka. Izbacili su koncept modula i uveli pakete kakve danas znamo, koji se oslanjaju na fajl sistem i uveli formalno koncept klasa iz objektno-orijentisane paradigme. Osim toga, jezik ima sintaksu sličnu jezicima C i C++, ali je mnogo stroži pri prevođenju, dizajniran tako da bude nezavisan od platforme, i sa pojednostavljenim upravljanjem memorijom. Pretpostavlja se da je ovo urađeno zbog popularnosti jezika C, ali i zbog jednostavnosti nekih struktura. Prva verzija je zvanično objavljena 1995. godine.
Java je, uz Kotlin, zvanično podržan jezik za izradu mobilnih aplikacija za Android uređaje.

XML:

XML je standardni skup pravila za definisanje formata podataka u elektronskoj formi. Propisan je od strane W3C. Sledeći pravila XML standarda, korisnici definišu sopstvene (XML) formate podataka, koje mogu koristiti za njihovo skladištenje, obradu i razmenu.

XML je skraćenica za Extensible Markup Language, odnosno proširivi (meta) jezik za označavanje (engl. markup) tekstualnih dokumenata. Ideja je bila da se stvori jezik koji će i ljudi i računarski programi moći jednostavno da čitaju. XML definiše opštu sintaksu za označavanje podataka pomoću odgovarajućih etiketa (engl. tags) koje imaju poznato ili lako razumljivo značenje. Format koji obezbeđuje XML za računarske elemente može se prilagoditi najrazličitijim oblastima, kao što su elektronska razmena podataka, čuvanje podataka, odvajanje podataka od prezentacije, vektorska grafika, sistemi glasovne pošte, izrada novih specijalizovanih jezika za označavanje.

Izvorni kod aplikacije i njeni zahtevi:

Dozvole koje aplikacija zahteva od uređaja da bi normalno funkcionisala se dodaju u *AndroidManifest.xml* i to:

```
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission android:name="android.permission.INTERNET" />
```

```
import androidx.appcompat.app.AppCompatActivity;

import android.annotation.SuppressLint;
import android.os.Bundle;
import android.webkit.WebSettings;
import android.webkit.WebView;
import android.webkit.WebViewClient;
```

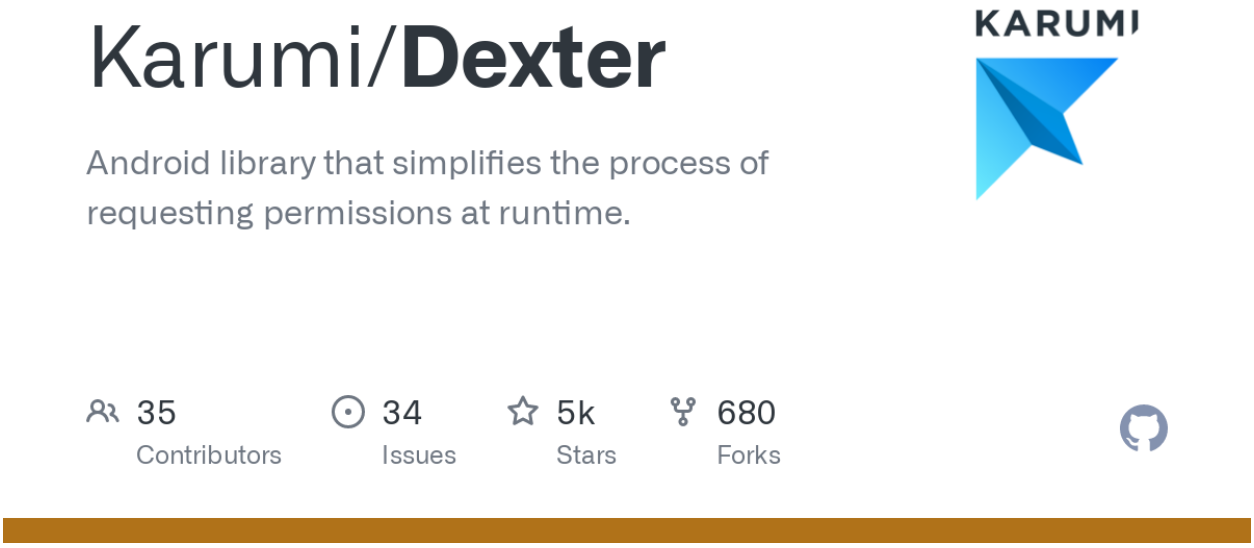
```
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.Manifest;
import android.content.Intent;
import android.os.Bundle;
import android.os.Environment;
import android.text.Editable;
import android.text.TextWatcher;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Adapter;
import android.widget.AdapterView;
import android.widget.ArrayAdapter;
import android.widget.BaseAdapter;
import android.widget.EditText;
import android.widget.ListView;
import android.widget.SearchView;
import android.widget.TextView;

import com.karumi.dexter.Dexter;
import com.karumi.dexter.PermissionToken;
import com.karumi.dexter.listener.PermissionDeniedResponse;
import com.karumi.dexter.listener.PermissionGrantedResponse;
import com.karumi.dexter.listener.PermissionRequest;
import com.karumi.dexter.listener.single.PermissionListener;

import java.io.File;
import java.util.ArrayList;
```

```
import androidx.annotation.NonNull;
import androidx.appcompat.app.AppCompatActivity;

import android.animation.AnimatorSet;
import android.animation.ObjectAnimator;
import android.content.Intent;
import android.graphics.PorterDuff;
import android.media.MediaPlayer;
import android.net.Uri;
import android.os.Bundle;
import android.os.Handler;
import android.view.MenuItem;
import android.view.View;
import android.widget.Button;
import android.widget.ImageView;
import android.widget.SeekBar;
import android.widget.TextView;

import java.io.File;
import java.util.ArrayList;
```

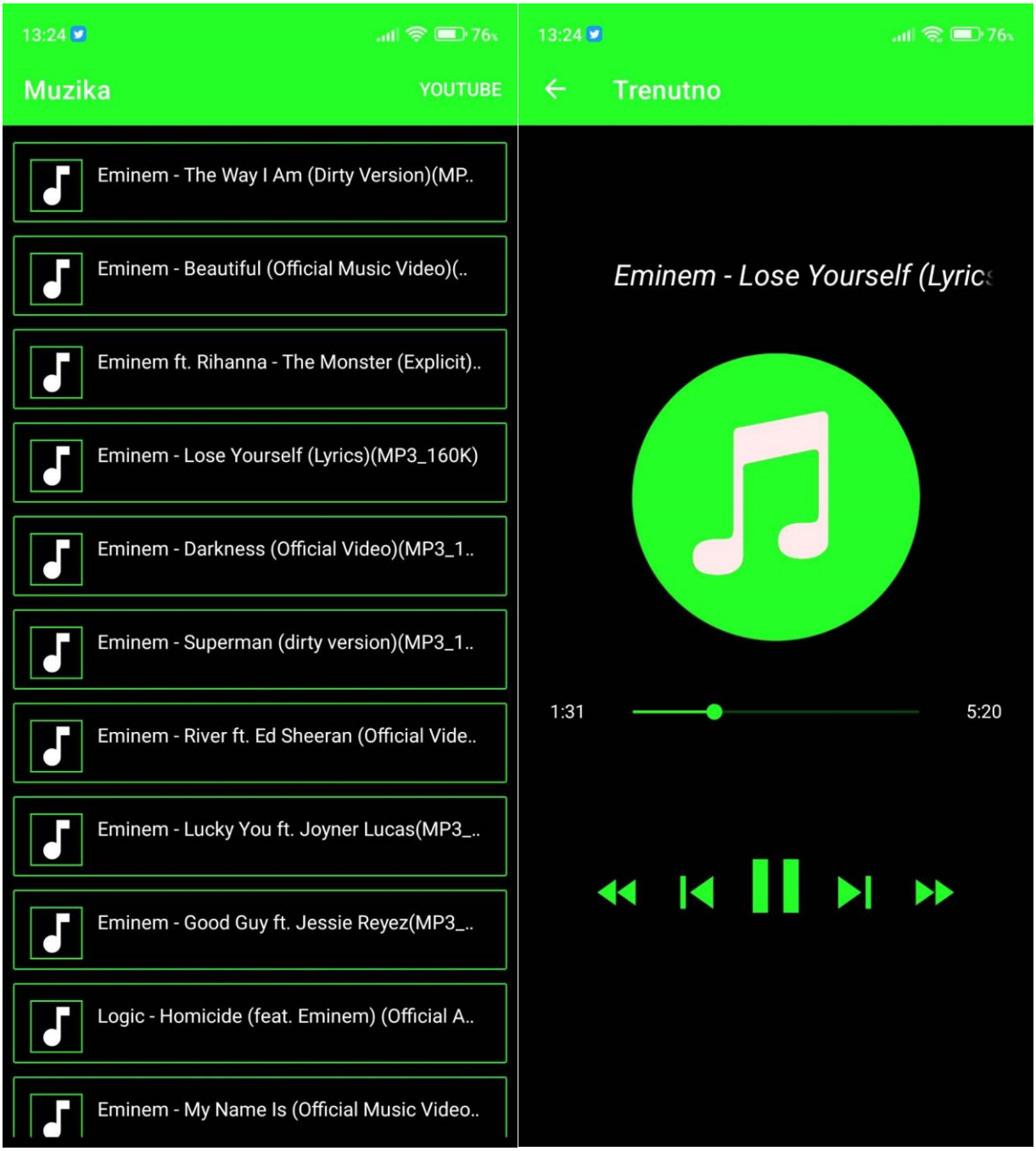
Opis korišćenih biblioteka:

Karumi Dexter:

Dekter je Android biblioteka koja pojednostavljuje postupak traženja dopuštenja tokom izvođenja.
Android Marshmallov uključuje novu funkcionalnost koja korisnicima omogućava davanje ili uključivanje odobrenja prilikom pokretanja aplikacija umesto da ih daje sve prilikom instalacije. Ovaj pristup daje korisniku veću kontrolu nad aplikacijama, ali zahteva od programera da doda puno koda za njegovu podršku.
Dekster oslobađa vaš kod dopuštanja vaših aktivnosti i omogućava vam pisanje logike gde god želite.



Izgled koriničkog interfejsa aplikacije:



Pokretanje youtube-a iz aplikcije (izvorni kod):

Korišćen je element WebView koji postoji u paleti elemenata Android studia. Dodat mu je osnovni url (youtube.com).

```
6 usages
WebView webView;

@SuppressLint("SetJavaScriptEnabled")

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main3);

    webView = findViewById(R.id.youtube);
    webView.setWebViewClient(new WebViewClient());
    webView.loadUrl("http://youtube.com");

    WebSettings webSettings = webView.getSettings();
    webSettings.setJavaScriptEnabled(true);
}

@Override
public void onBackPressed() {
    if(webView.canGoBack()) {
        webView.goBack();
    } else {
    super.onBackPressed(); }
}
}
```

Funkcionalnosti aplikacije:

```
getSupportActionBar().setTitle("Trenutno"); //title koji se nalazi u headeru
getSupportActionBar().setDisplayHomeAsUpEnabled(true);
getSupportActionBar().setDisplayShowHomeEnabled(true);

previous = findViewById(R.id.previous);
next = findViewById(R.id.next);
play = findViewById(R.id.play);
rewind = findViewById(R.id.rewind);
fastforward = findViewById(R.id.fastforward);
textStart = findViewById(R.id.textStart);
textStop = findViewById(R.id.textStop);
textSongName = findViewById(R.id.textSongName);
seekBar = findViewById(R.id.seekbar);
imageview = findViewById(R.id.imageview);
```

Ažuriraj seekbar:

```
azurirajseekbar = (Thread) run() -> {
    int trajanje = mediaPlayer.getDuration();
    int trenutno = 0;

    while (trenutno < trajanje) {
        try {
            sleep( millis: 500);
            trenutno = mediaPlayer.getCurrentPosition();
            seekBar.setProgress(trenutno);
            }
        catch (InterruptedException | IllegalStateException e)
        {
            e.printStackTrace();
        }
        }
};

seekBar.setMax(mediaPlayer.getDuration());

azurirajseekbar.start();
```

Seekbar:

```
seekBar.getProgressDrawable().setColorFilter(getResources().getColor(R.color.primary), PorterDuff.Mode.MULTIPLY);
seekBar.getThumb().setColorFilter(getResources().getColor(R.color.primary), PorterDuff.Mode.SRC_IN);

seekBar.setOnSeekBarChangeListener(new SeekBar.OnSeekBarChangeListener() {
    @Override
    public void onProgressChanged(SeekBar seekBar, int progress, boolean fromUser) {

    }

    @Override
    public void onStartTrackingTouch(SeekBar seekBar) {

    }

    @Override
    public void onStopTrackingTouch(SeekBar seekBar) {
        mediaPlayer.seekTo(seekBar.getProgress());

    }
});

String trajanjePesme = ukupnoVreme(mediaPlayer.getDuration());
textStop.setText(trajanjePesme);

final Handler handler = new Handler();
final int delay = 1000;

handler.postDelayed(new Runnable() {
    @Override
    public void run() {
        String trenutno = ukupnoVreme(mediaPlayer.getCurrentPosition());
        textStart.setText(trenutno);
        handler.postDelayed( this, delay);
    }
}, delay);
```

Play / Pause:

```
play.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View view) {
        if (mediaPlayer.isPlaying())
        {
            play.setBackgroundResource(R.drawable.ic_play);
            mediaPlayer.pause();
        } else {
            play.setBackgroundResource(R.drawable.ic_pause);
            mediaPlayer.start();
        }
    }
});
```

Next:

```
mediaPlayer.setOnCompletionListener(new MediaPlayer.OnCompletionListener() {
    @Override
    public void onCompletion(MediaPlayer mp) { next.performClick(); }
});

next.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mediaPlayer.stop();
        mediaPlayer.release();
        position = ((position+1)%mySongs.size());
        Uri u = Uri.parse(mySongs.get(position).toString());
        mediaPlayer = MediaPlayer.create(getApplicationContext(), u);
        songName[0] = mySongs.get(position).getName();
        textSongName.setText(songName[0]);
        mediaPlayer.start();
        play.setBackgroundResource(R.drawable.ic_pause);
        Animacija(imageview);
    }
});
```

Previous:

```
previous.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mediaPlayer.stop();
        mediaPlayer.release();
        position = ((position-1)<0)?(mySongs.size()-1):(position-1);
        Uri u = Uri.parse(mySongs.get(position).toString());
        mediaPlayer = MediaPlayer.create(getApplicationContext(), u);
        songName[0] = mySongs.get(position).getName();
        textSongName.setText(songName[0]);
        mediaPlayer.start();
        play.setBackgroundResource(R.drawable.ic_pause);
        Animacija(imageview);
    }
});
```

FastForward / Rewind:

```
fastforward.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        if(mediaPlayer.isPlaying())
        {
            mediaPlayer.seekTo( msec: mediaPlayer.getCurrentPosition()+20000);
        }
    }
});

rewind.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        mediaPlayer.seekTo( msec: mediaPlayer.getCurrentPosition()-20000);
    }
});
```

Duration:

```
2 usages
public String ukupnoVreme (int duration)
{
    String vreme = "";
    int min = duration/1000/60;
    int sec = duration/1000%60;

    vreme+=min+":";
    if(sec<10) {
        vreme+="0";
    }
    vreme+=sec;
    return vreme;
}
```

Animacija sličice na sredini (Muzička nota):

```
2 usages
public void Animacija (View view) {
    ObjectAnimator animator = ObjectAnimator.ofFloat(imageview,  propertyName: "rotation",  ...values: 0f, 360f);
    animator.setDuration(1500);
    AnimatorSet animatorset = new AnimatorSet();
    animatorset.playTogether(animator);
    animatorset.start();
}
```

Na klik korisnika na *Next* dugme (promena na narednu pesmu), dugme izvršava animaciju rotacije za 360 stepeni. Takođe, animacije se izvršava i kada se pesma sama završi do kraja i pređe na novu pesmu.

Literatura:

Moodle portal: Programiranje mobilnih aplikacija:
http://moodle.fink.rs/course/view.php?id=982

Wikipedia: Android studio:
https://en.wikipedia.org/wiki/Android_Studio
https://en.wikipedia.org/wiki/Java_(programming_language)
https://en.wikipedia.org/wiki/XML

GitHub: Karumi Dexter:
https://github.com/Karumi/Dexter