

Programiranje mobilnih uređaja

4. razred

Željko Turkalj, prof. mat. i inf.

Tehnička škola Kutina

1 Višestruki dodiri zaslona

1.1 XML

U ovom dijelu jedan je od najjednostavniji xml dokumenata. Za pokazati kako rade višestruki dodiri potrebno je jedino zaslonu postaviti id.

```
<?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:id="@+id/ekran">

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

1.2 Java

U java kodu koristimo već poznatu metodu *setOnTouchListener* kada želim imati interakciju sa cijelim zaslonom.

```
public class MainActivity extends AppCompatActivity {

    ConstraintLayout zaslon;

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

        zaslon= (ConstraintLayout)findViewById(R.id.ekran);

        zaslon.setOnTouchListener(new View.OnTouchListener() {
            @Override
            public boolean onTouch(View v, MotionEvent event) {

                int stisnuto=event.getActionMasked();

                switch (stisnuto) {

                    case MotionEvent.ACTION_POINTER_DOWN:
                        Log.i("Ispis", "Broj
prstiju"+event.getPointerCount());
```

```
        break;

        case MotionEvent.ACTION_DOWN:
            Log.i("Ispis", "Prst na zaslonu");
            break;
        case MotionEvent.ACTION_UP:
            Log.i("Ispis", "Prst dignut");
            break;
        case MotionEvent.ACTION_MOVE:
            Log.i("Ispis", "Prst se mice");
            break;
    }
    /* Standardno je return false, ali to moramo pretvoriti u true ako hoćemo
    da se metoda setOnTouchListener ponaša kao da je u petlji unutar vremena
    odvijanja aplikacije te da se stalno osvježava MotionEvent*/
    return true;
}
});
}
}
```

Projektni zadatak:

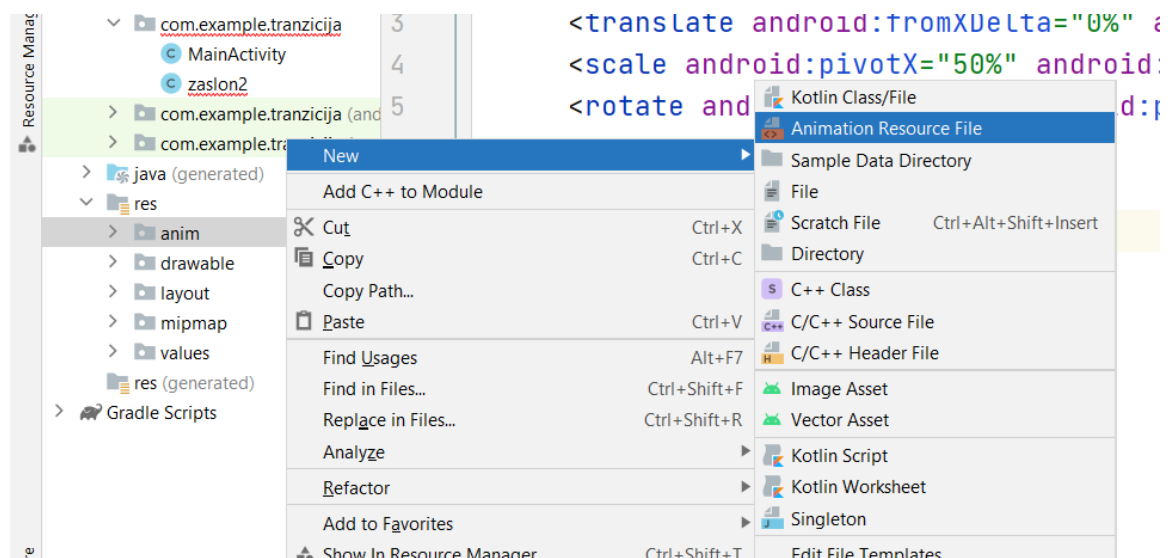
1. Napravi jednostavnu aplikaciju koja radi sljedeće:

Početni ekran je bijel, ako je prst na ekranu podloga se mijenja u crvenu, ako se prst miče podloga je plava, ako se prst makne podloga se vraća na bijelu. Ako su dva prsta na ekranu pojavljuje se veliki broj 2. Ako su tri prsta na ekranu reproducira se zvuk. Ako su četiri prsta na ekranu prelazimo na novi ekran na kojem moramo staviti svih pet prstiju za povratak natrag. Može i nešto drugo sa raznim varijacijama na temu od onoga što smo do sada naučili.

2 Animacija

Animirati možemo razne widgete koje koristimo pri kreiranju aplikacije, ali moramo biti oprezni da animacije ne odvuču pažnju, nego naglase baš onaj dio aplikacije koji treba uspostaviti interakciju sa korisnikom. Tako na primjer ako moramo unijeti nekakav broj možemo animirati edit tekst koji se bude malo povećao. Nakon toga možemo povećati gumb itd.

Kreiranjem projekta za izradu aplikacija već smo naučili da se kreiraju najbitnije mape i datoteke važne u izradi aplikacije: manifest, java, res. Obzirom da se animacije također pišu u xml kodu kao i izgled aplikacije potrebna je još jedna mapa unutar resursa koja nije odmah integrirana, nego ju moramo napraviti naknadno. To napravimo kreiranjem nove mape resursa koju moramo nazvati anim i tipa je anim. Nakon toga unutar te mape kreiramo novi „Animation resource file“.



Kod za napraviti animacije je vrlo jednostavan. Potrebno je naučiti samo nekoliko tagova koji omogućavaju translaciju, rotaciju, skaliranje i prozirnost.

translate – translacija

Osnovni atributi su: FromXDelta, ToXDelta, FromYDelta, ToYDelta

Ako koristimo:

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="1000" android:repeatMode="restart" >

    <translate android:fromXDelta="0%" android:toXDelta="20%"/>

</set>
```

0% znači da animacija kreće sa mjesta gdje je objekt postavljen, a 20% da će se pomaknuti udesno sve dok se pozicija po x-su ne poveća za 20% širine ekrana.

Ako umjesto 0% i 20% koristimo 0dp i 20dp to bi značilo da se objekt bude počeo kretati sa lijevog ruba zaslona i pomaknuti će se 20dp udesno.

Naravno uvijek se moramo pitati kod animacija koliko će one trajati. To se postavlja u tagu <set>. Atribut je **duration** i postavlja se u mili sekundama. Također animaciju možemo ponavljati ili ju odraditi samo jednom. Atribut taga **set** za to je **repeatMode** koji možemo postaviti na **repeat** ili **reverse**.

Rotacija, skaliranje i prozirnost imaju slične tagove i attribute. Prouči ih.

Za pokretanje animacije moramo prvo kreirati objekt klase Animation:

```
Animation nasaAnimacija;

nasaAnimacija = AnimationUtils.loadAnimation(getApplicationContext(),
    R.anim.proba);
```

Skoro svi widgeti imaju metodu startAnimation(). Pozivanjem ove metode pokreće se animacija.

```
sampleTextView.startAnimation(nasaAnimacija);
```

Zadatak:

Napravi aplikaciju koja ima jedan TextView na kojem piše današnji datum u gornjem lijevom kutu zaslona. Klikom bilo gdje na zaslon tekst se kreće prema donjem desnom kutu zaslona. Povećava se 50% i rotira se za 360°.

Napomena: Za rješavanje ovog problema probaj postavljati različitim redoslijedom tagove za translaciju, rotaciju i skaliranje unutar anim dokumenta.

3 Tranzicija između dva ekrana

Pojam tranzicija u programiranju aplikacija koristimo za animiranje prijelaza između dva zaslona.

Zadatak:

Napravite dva zaslona. Na prvom kreirajte gumb koji omogućava odlazak na drugi zaslon, a na drugom omogućite da pomicanje prsta po zaslonu aktivira povratak na početni zaslon.

Nakon testiranja ovog zadatka i prijelaza između dva zaslona bez grešaka idemo napraviti kodove tranzicija. Oni se rade u mapi **anim** koju kreiramo u mapi **Resource**.

slide_from_bottom.xml

```
<!--slide from bottom-->
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="300">
    <translate android:fromYDelta="100%"
        android:toYDelta="0%"/>
</set>
```

slide_to_top.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="300">
    <translate
        android:fromYDelta="0%"
        android:toYDelta="-100%"/>
</set>
```

slide_from_left.xml

```
<?xml version="1.0" encoding="utf-8"?>
```

```
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="300">
    <translate
        android:fromXDelta="-100%"
        android:toXDelta="0%"/>
</set>
```

slide_from_right.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="300">
    <translate
        android:fromXDelta="100%"
        android:toXDelta="0%"/>
</set>
```

zoom_in.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="300">
    <scale
        android:fromXScale="0.8"
        android:fromYScale="0.8"
        android:pivotX="50%"
        android:pivotY="50%"
        android:toXScale="1.0%"
        android:toYScale="1.0%" />
    <alpha
        android:fromAlpha="0"
        android:toAlpha="1" />
</set>
```

zoom_out.xml

```
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="500">
    <scale
        android:fromXScale="1.0"
```

```
        android:fromYScale="1.0"
        android:pivotX="50%"
        android:pivotY="50%"
        android:toXScale="0.8%"
        android:toYScale="0.8%" />
    <alpha
        android:fromAlpha="1"
        android:toAlpha="0"/>
</set>

static_animation.xml:
<?xml version="1.0" encoding="utf-8"?>
<set xmlns:android="http://schemas.android.com/apk/res/android"
    android:duration="300">
</set>

package com.android.gfgapplication;

import androidx.appcompat.app.AppCompatActivity;

import android.content.Intent;
import android.os.Bundle;
import android.view.View;

public class MainActivity extends AppCompatActivity {

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

        findViewById(R.id.buttonOpenSecondActivity).setOnClickListener(new
View.OnClickListener() {
            @Override
            public void onClick(View view) {
                startActivity(new
Intent(MainActivity.this, SecondActivity.class));
            }
        });
    }
}
```

```
        // for right to left animation
        //
        overridePendingTransition(R.anim.slide_from_right,R.anim.slide_to_left);

        // for bottom to top animation
        //
        overridePendingTransition(R.anim.slide_from_bottom,R.anim.slide_to_top);

        // for top to bottom animation
        //
        overridePendingTransition(R.anim.slide_from_top,R.anim.slide_to_bottom);

        // zoom-in animation
        //
        overridePendingTransition(R.anim.zoom_in,R.anim.static_animation);

        // zoom-out animation
        //
        overridePendingTransition(R.anim.static_animation,R.anim.zoom_out);
    }

    });
}

}
```

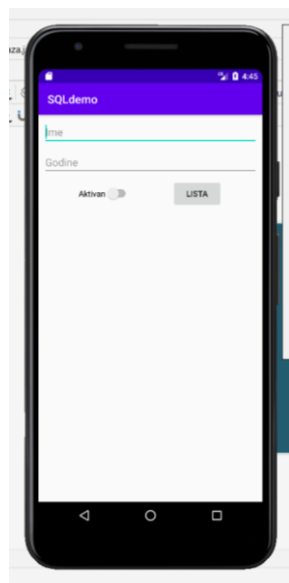
4 BAZE

4.1 XML

Kreiramo xml dokument koji omogućuje unos jednog korisnika i njegovih podataka ime, godine, aktivan.

Dakle potreban nam je jedan EditText za unos imena, jedan EditText za unos godina, jedan Switcher za provjeru aktivnost, jedan Button za unos i pregled vrijednosti iz baze podataka te ListView za prikaz podataka iz baze.

Unos imena i godina se trebaju poklapati po širini sa zaslonom, te imati margine od 8px. Switcher i Button neka budu jednoliko pozicionirani ispod unosa. Ispod svega dolazi lista koja je poravnata lijevo, desno na zaslon i gore na gumb.



```
<?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"
tools:context=".MainActivity"
tools:ignore="ExtraText">

<EditText
    android:id="@+id/et_ime"
    android:layout_width="match_parent"
    android:layout_height="wrap_content"
    android:layout_marginStart="8dp"
```

```

        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:autoFillHints=""
        android:hint="@string/korisnik_ime"
        android:inputType="textPersonName"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent" />

```

<EditText

```

        android:id="@+id/et_godine"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:layout_marginStart="8dp"
        android:layout_marginTop="8dp"
        android:layout_marginEnd="8dp"
        android:inputType="number"
        android:hint="@string/korisnikgodine"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/et_ime"
    />

```

<Switch

```

        android:id="@+id/sw_aktivan"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="@string/korisnik_aktivnost"
        app:layout_constraintEnd_toStartOf="@+id/btn_unesi"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="@+id/btn_unesi"
        app:layout_constraintBottom_toBottomOf="@+id/btn_unesi"
        tools:ignore="UseSwitchCompatOrMaterialXml" />

```

<Button

```

        android:id="@+id/btn_unesi"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_marginTop="8dp"
        android:text="@string/lista"
        app:layout_constraintStart_toEndOf="@+id/sw_aktivan"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintTop_toBottomOf="@+id/et_godine" />

```

<ListView

```

        android:id="@+id/lw_lista"
        android:layout_width="409dp"
        android:layout_height="532dp"
        android:layout_marginTop="220dp"
        app:layout_constraintEnd_toEndOf="parent"
        app:layout_constraintStart_toStartOf="parent"
        app:layout_constraintTop_toTopOf="parent">

```

</ListView>

</androidx.constraintlayout.widget.ConstraintLayout>

4.2 Kreiranje klase ModelKorisnika

Nakon toga kreirajmo jednu klasu koja opisuje korisnika. Obzirom da želimo jedan parametar koji će kod svakog korisnika biti različit koristiti ćemo id tipa int, ime tipa String, godine tipa int i aktivan tipa boolean. Napravi dva konstruktora, jedan prazan i jedan sa mogućnosti unosa sva četiri parametra. Zatim sve gettere i settere, metodu to string.

```
package com.example.sqldemo;

class ModelKorisnika {

    private int id;
    private String ime;
    private int godine;
    private boolean aktivan;

    //kreiranje konstruktora

    public ModelKorisnika(int id, String ime, int godine, boolean aktivan)
    {
        this.id = id;
        this.ime = ime;
        this.godine = godine;
        this.aktivan = aktivan;
    }

    public ModelKorisnika() {
    }
    // kreiranje getters and setters

    public int getId() {
        return id;
    }

    public String getIme() {
        return ime;
    }

    public int getGodine() {
        return godine;
    }

    public boolean isAktivan() {
        return aktivan;
    }

    public void setId(int id) {
        this.id = id;
    }

    public void setIme(String ime) {
```

```

        this.ime = ime;
    }

    public void setGodine(int godine) {
        this.godine = godine;
    }

    public void setAktivan(boolean aktivan) {
        this.aktivan = aktivan;
    }

    // metoda za ispis u string

    @Override
    public String toString() {
        return "id=" + id +
            ", ime='" + ime + '\'' +
            ", godine=" + godine +
            ", aktivan=" + aktivan;
    }
}

```

4.3 Početni java kod

Povežimo java kod sa xml-om i klasom. Pokušaj unijeti podatke u model te ih ispisati u Toust poruci. Pazi što konstruktor modela korisnika traži, id možemo staviti na -1, string za ime nije problem, ali godine su tipa int pa moramo se prisjetiti kako se broj koji je zapisan u stringu pretvara u cijeli broj (Integer.parseInt), kod switcha ne uzimamo string nego provjeravamo uključenost metodom isChecked(). Kompletan java do ovog koraka glasi:

```

public class MainActivity extends AppCompatActivity {

    Button btn_dodaj;
    Switch sw_aktivan;
    ListView lv_lista;
    EditText et_ime, et_godine;
    ModelKorisnika modelKorisnika;

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

        btn_dodaj=(Button) findViewById(R.id.btn_unesi);
        sw_aktivan=(Switch) findViewById(R.id.sw_aktivan);
        lv_lista=(ListView) findViewById(R.id.lw_lista);
        et_godine=(EditText) findViewById(R.id.et_godine);
        et_ime=(EditText) findViewById(R.id.et_ime);
    }
}

```

```

        btn_dodaj.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                try {
                    modelKorisnika=new ModelKorisnika(-1,
et_ime.getText().toString(),
Integer.parseInt(et_godine.getText().toString()), sw_aktivan.isChecked());
                    Toast.makeText(MainActivity.this,
modelKorisnika.toString(), Toast.LENGTH_SHORT).show();
                }
                catch (Exception e){
                    Toast.makeText(MainActivity.this, "Nemoguće kreirati
korisnika bez godina starosti", Toast.LENGTH_SHORT).show();
                }
            }
        });
    }
}

```

4.4 Kreirajmo BAZU

Baza se kreira uobičajenim postupkom stvaranja klase. Nakon što je napravljena klasa dodajemo iza naziva klase extends SQLiteOpenHelper

```

class Baza extends SQLiteOpenHelper{
}

```

Vidimo da se sada javlja greška, jer SQL zahtijeva neke metode. Iskoristimo prijedlog Android Studia za kreiranjem metoda i kreiramo dvije metode:

```

class Baza extends SQLiteOpenHelper{
    @Override
    public void onCreate(SQLiteDatabase db) {

    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {

    }
}

```

Vidimo da se još uvijek javlja greška. Android studio predlaže kreiranje konstruktora. Odaberemo prijedlog.

```

class Baza extends SQLiteOpenHelper{
    public Baza(@Nullable Context context, @Nullable String name, @Nullable
SQLiteDatabase.CursorFactory factory, int version) {
        super(context, name, factory, version);
    }
}

```

```

@Override
public void onCreate(SQLiteDatabase db) {

}

@Override
public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {

}
}

```

Više se ne javlja greška, ali naš konstruktor ima puno parametara pa ćemo ih malo reducirati i unijeti neke ručno. Ostavimo samo context. Metodi *super* upišemo parametre: name postavim na recimo baza, factory na null, a version na 1.

```

class Baza extends SQLiteOpenHelper{
    public Baza(@Nullable Context context) {
        super(context, "baza", null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {

    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {

    }
}

```

Krenimo sada sa kreiranjem tablice baze i njenih stupaca. Za kreiranje baze koristimo standardni SQLite.

Obzirom da ovaj kod mora biti string kako bismo ga mogli pozvati metodom `execSQL` objekta `db` koji je argument u našoj metodi `OnCreate` za početak napravimo jedan prazan string i pozovemo metodu `execSQL` objekta `db`.

```

@Override
public void onCreate(SQLiteDatabase db) {

    String kreiraj_tablicu="";
    db.execSQL(kreiraj_tablicu);

}

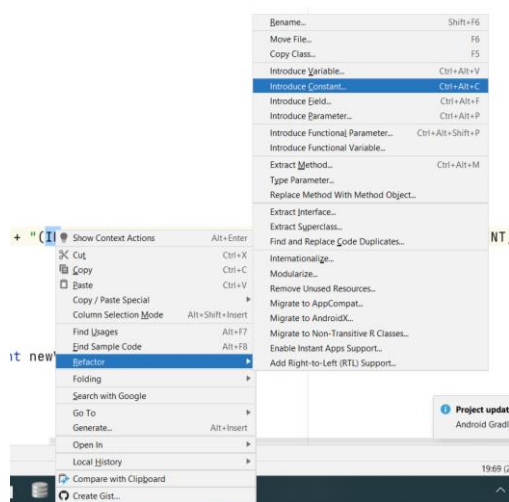
```

Sada će nam prilikom kreiranja stringa za kreiranje SQLite tablice Android Studio pomoći i ukazivati nam na ispravnost koda. Unesimo kod za kreiranje tablice.

```
@Override
public void onCreate(SQLiteDatabase db) {

    String kreiraj_tablicu="CREATE TABLE MojaTablica(ID INTEGER PRIMARY KEY
    AUTOINCREMENT, IME TEXT, GODINE INT, AKTIVAN BOOL)";
    db.execSQL(kreiraj_tablicu);
}
```

Pošto ćemo morati moći dohvatiti naziv naše tablice i sve stupce napravimo konstante tako da ih redom obilježavamo i desni klik na njih, te odaberemo *Refactor->Introduce Constant...* (Svim stupcima tablice dodajemo ispred naziva Stupac tako da se kasnije lakše snalazimo u kodu)



Sada naš kod izgleda ovako:

```
package com.example.sqldemo;

import android.content.Context;
import android.database.sqlite.SQLiteDatabase;
import android.database.sqlite.SQLiteOpenHelper;

import androidx.annotation.Nullable;

class Baza extends SQLiteOpenHelper{
    public static final String MOJA_TABLICA = "MojaTablica";
    public static final String Stupac_ID = "ID";
    public static final String Stupac_IME = "IME";
    public static final String Stupac_GODINE = "GODINE";
    public static final String Stupac_AKTIVAN = "AKTIVAN";

    public Baza(@Nullable Context context) {
        super(context, "baza", null, 1);
    }

    @Override
    public void onCreate(SQLiteDatabase db) {

        String kreiraj_tablicu= "CREATE TABLE " + MOJA_TABLICA + " (" +
```

```

        Stupac_ID + " INTEGER PRIMARY KEY AUTOINCREMENT, " + Stupac_IME + " TEXT, "
        + Stupac_GODINE + " INT, " + Stupac_AKTIVAN + " BOOL)";
        db.execSQL(kreiraj_tablicu);

    }

    @Override
    public void onUpgrade(SQLiteDatabase db, int oldVersion, int
newVersion) {

    }

}

```

4.5 Unos korisnika

Napravimo metodu za unos jednog korisnika:

```

public boolean dodaj(ModelKorisnika korisnik){
    //koristimo klasu SQLiteDatabase kako bismo kreirali bazu u koju možemo
pisati
    SQLiteDatabase db=this.getWritableDatabase();
    // ContentValues je klasa kojom kreiramo objekt kojim definiramo što se
upisuje u koji stupac naše tablice. Za to koristimo metodu put
    ContentValues unos=new ContentValues();
    unos.put(Stupac_IME, korisnik.getIme());
    unos.put(Stupac_GODINE, korisnik.getGodine());
    unos.put(Stupac_AKTIVAN, korisnik.isAktivan());
    // Nakon što smo kreirali ContentValues metodom insert postavljamo
vrijednosti u tablicu
    // metoda vraća vrijednost long, prvi argument joj je ime tablice,
drugi null, a treći objekt klase ContentValues
    long test=db.insert(MOJA_TABLICA,null,unos);
    // ako insert vrati -1 nisu unesene vrijednosti u tablicu pa napravimo
provjeru
    if(test== -1)
    {
        return false;
    }
    else{
        return true;
    }

}

```

Kada smo napravili metodu za unos idemo probati unijeti korisnika u bazu. U MainActivity.java pozovemo metodu *dodaj()*.

```

btn_dodaj.setOnClickListener(new View.OnClickListener() {
    @Override
    public void onClick(View v) {
        try {
            modelKorisnika=new ModelKorisnika(-1,
            et_ime.getText().toString(),
            Integer.parseInt(et_godine.getText().toString()), sw_aktivan.isChecked());
            Toast.makeText(MainActivity.this, modelKorisnika.toString(),
            Toast.LENGTH_SHORT).show();
        }
    }
}

```

```

        catch (Exception e){
            Toast.makeText(MainActivity.this, "Nemoguće kreirati korisnika
bez godina starosti", Toast.LENGTH_SHORT).show();
        }

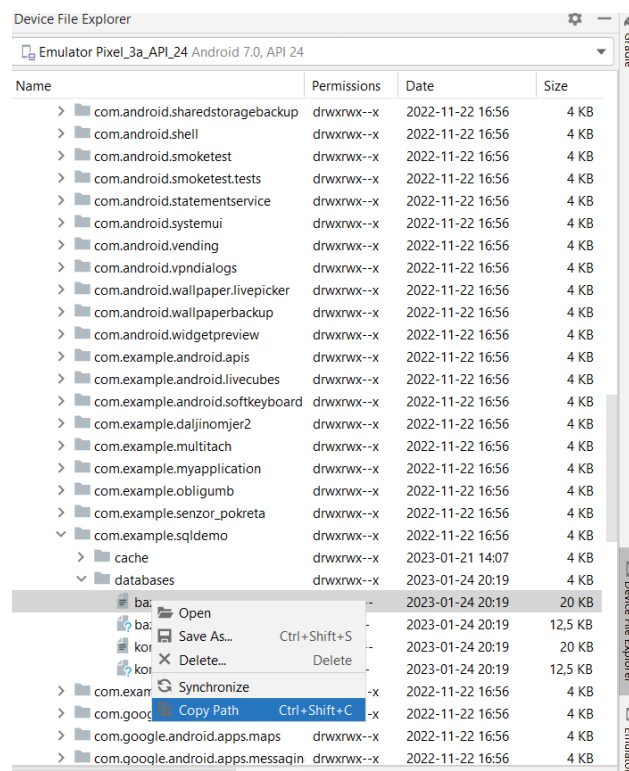
//Definiramo objekt baza koji je tipa klase Baza

        baza = new Baza(MainActivity.this);
        boolean test = baza.dodaj(modelKorisnika);
        Toast.makeText(MainActivity.this, "Test" + test,
Toast.LENGTH_SHORT).show();

    }
});

```

Ako je sve u redu tablica sa vrijednostima baze biti će kreirana, ali vrijednost još nisu vidljive. **Gdje je tablica baze?** Nalazi se negdje na mobilnom uređaju ili na emulatoru. Da bismo vidjeli podatke na emulatoru on mora biti upaljen i idemo u Android Studiu na *View->Tool windows->Device File Explorer*. Sada tražimo *data->data->ime našeg projekta->database*. Ako je sve u redu ovdje bi se trebala nalaziti naša baza. Kako bismo provjerili podatke koji su upisani u nju možemo napraviti *copy path* i prekopirati u neku mapu našeg računala, te ju zatim otvoriti nekim od programa za čitanje SQLite baze.



4.6 Dohvaćanje podataka iz tablice

Napravimo sada metodu za dohvatiti podatke iz baze:

```
//metoda treba vratiti listu korisnika
public List<ModelKorisnika> dohvatiBazu(){
    //Lista mora vratiti više korisnika koji su zapisanu svaki u svom redu
    //tablice baze
    List<ModelKorisnika> povratn_lista = new ArrayList<>();
    //Kreiramo string za čitanje iz baze
    String citaj_tablicu = "SELECT * FROM "+ MOJA_TABLICA;
    // kreiramo objekt za čitanje iz baze
    SQLiteDatabase db=this.getReadableDatabase();
    //Slično kao kod miša trebamo nešto što će pokazivati na svaki red tablice,
    //vidi čuda zovemo ga:
    Cursor cursor=db.rawQuery(citaj_tablicu,null);
    //Pomaknemo cursor na prvi red i čitamo sve dok ne dođe do zadnjeg,
    //pripazimo na to da numeracija u SQLite tablicama po stupcima ide od 0,
    //druga zamka je podatak tipa boolean. Iako je u tablici on boolean ili bool
    //on je u SQLite tablici 0 ili 1, zato radimo jedan if else.
    if(cursor.moveToFirst()){
        do{
            int IDKorisnika=cursor.getInt(0);
            String ImeKorisnika=cursor.getString(1);

            int GodineKorisnika=cursor.getInt(2);
            boolean korisnikAktivan=true;
            if(cursor.getInt(3) == 1) {
                korisnikAktivan = true;
            } else
                korisnikAktivan = false;
            //od svakog reda tablice radimo jednog korisnika pomoću našeg konstruktora
            ModelKorisnika korisnici_po_redovima=new
            ModelKorisnika(IDKorisnika,ImeKorisnika,GodineKorisnika,korisnikAktivan);
            //svakog korisnika dodajemo u našu listu metodom add
            povratn_lista.add(korisnici_po_redovima);
        }while(cursor.moveToNext());
    }
    //zatvorimo cursor i bazu
    cursor.close();
    db.close();
    //😊
    return povratn_lista;
}
```

4.7 ListView MainActivity.java

Kako bismo sada ispisali podatke naše tablice u ListView trebamo dodati android.widget:

```
ArrayAdapter bazaKorisnikaLista;
```

I još malo koda:

```
bazaKorisnikaLista = new ArrayAdapter<ModelKorisnika>(MainActivity.this,
android.R.layout.simple_list_item_1, baza.dohvatiBazu());
lv_lista.setAdapter(bazaKorisnikaLista);
```

Zadaci za vježbu

Zad. 1. Primijetimo da naša aplikacija ne učitava bazu dok ne unesemo novog korisnika. Zbog toga ispišimo bazu odmah prilikom pokretanja aplikacije.

Zad. 2. Promijeni string unutar metode dohvatiBazu tako da se elementi ispisuju:

a) poredani abecednim redom obzirom na ime,

b) poredani od najmlađeg do najstarijeg,

c) ispisuju se samo aktivni korisnici

Zad. 3. Obzirom da se dio koda ponavlja prilikom učitavanja baze i dodavanja novog korisnika, napravi od tog dijela metodu unutar *MainActivity.java* koja će zamijeniti:

```
bazaKorisnikaLista = new ArrayAdapter<ModelKorisnika>(MainActivity.this,
android.R.layout.simple_list_item_1, baza.dohvatiBazu());
lv_lista.setAdapter(bazaKorisnikaLista);
```

Nazovite ju *citanjeBaze()*.

4.8 Brisanje elemenata baze odabirom elementa u LISTI

Ovaj dio je jednostavan. Želimo obrisati korisnika pod određenim id-om. Napravimo bazu koja se može mijenjati. Upišemo string brisanja i pozovemo metodu *rawQuery* isto kako kod čitanja iz baze.

```
public void obrisiKorisnika(ModelKorisnika modelKorisnika)
{
    SQLiteDatabase db=this.getWritableDatabase();
    String stringBrisanja=" DELETE FROM " + CUSTOMER_TABLE + " WHERE " +
    ID + " = " + modelKorisnika.getId();
    db.rawQuery(stringBrisanja, null);
    db.close();
}
```

Idemo sada pozvati ovu funkciju odabirom korisnika iz LISTE. Metoda za odabir korisnika iz LISTE slična je metodi klika na gumb. Jedina je razlika što ovdje moramo dodati *Itam*. Dakle kod glasi:

```
bazaKorisnikaLista.setOnItemClickListener(new  
AdapterView.OnItemClickListener() {  
    @Override  
    public void onItemClick(AdapterView<?> parent, View view, int position,  
long id) {  
        ModelKorisnika odabranikorisnik = (ModelKorisnika)  
parent.getItemAtPosition(position);  
        dbHelper.obrisiKorisnika(odabranikorisnik);  
        citanjebaze();  
    }  
});
```

Zadaci za vježbu

1. Dodaj jedan RadioButton koji će omogućiti brisanje/nebrisanje iz baze odabirom korisnika. Ako je isključen odabirom korisnika ne događa se ništa, a ako je uključen korisnik se briše iz baze.
2. Istraži kako se dodaje skočni prozor. On treba imati pitanje za korisnika: Jeste li sigurni da želite obrisati korisnika?

Projektni zadatak

Napravi aplikaciju koja omogućava korisniku unos podataka o automobilu. Podatke izmisli sam, ali oni moraju sadržavati String, realne brojeve, cijele brojeve i istina/laž. Aplikacija sadrži više ekrana. Prvi je navigacijski, sa njega idemo na ekran koji omogućava unos podataka u bazu ili na drugi koji služi za ispis podataka iz baze različitim sortiranjem, te brisanje podataka o automobilima. Pokušaj se sjetiti što više stvari koje smo napravili u ove dvije godine i primjeni ih u ovoj aplikaciji (obli gumbi, animacija objekta, prijelaz između zaslona itd., reprodukcija zvuka). Svi podaci pohranjuju se u lokalnu bazu. Pokušaj kreirati bazu podataka sa dvije tablice koje su međusobno povezane podatkom ID. Projekt se radi na školskom računalu i svaki put se bilježi vaš napredak u stvaranju aplikacije.