АКАДЕМИЈА ТЕХНИЧКО-ВАСПИТАЧКИХ СТРУКОВНИХ СТУДИЈА
ОДСЕК НИШ

ANDROID PROGRAMIRANJE

Vežba 3: Razvoj jednostavne TouristInfo aplikacije

Teorijske napomene

Ova aplikacija će nuditi korisnicima bitne informacije o različitim turističkim lokacijama u Srbiji. Na glavnom ekranu će biti prikazana lista dostupnih kategorija, a zatim korisnik može da izabere jednu od njih i dobije detaljnije informacije, kao što su adresa, kontakt telefon ili radno vreme.

Ključne komponente koje ćemo za sada koristiti pri izradi su:

1. **Fragmenti**: Za pregled lista i detalje o turističkim mestima.
2. **Meniji**: Za promenu prikaza i filtriranje prikazanih mesta po kategoriji.
3. **Eventovi**: Obrada klikova i navigacija između različitih ekrana.

Osnovne funkcionalnosti aplikacije:

1. **Prikaz liste kategorija**: Na glavnom ekranu aplikacije prikazuje se lista kategorija, npr:
    - Istorijska mesta (tvrđave, parkovi, stari gradovi...),
    - Prirodne lepote (jezera, planine, nacionalni parkovi...),
    - Umetničke galerije i muzeji.
2. **Pregled detaljnih informacija**: Kada korisnik kategoriju, prikazuju mu se detaljniji podaci o svakoj lokaciji, uključujući:
    - **Naziv lokacije**: naziv znamenitosti ili lokacije.
    - **Opis**: kratak opis koji korisnicima pruža osnovne informacije o lokaciji.
    - **Kontakt telefon**: broj telefona za dodatne informacije.
    - **Radno vreme**: radno vreme lokacije, što olakšava korisnicima planiranje poseta.
3. **Meni opcije**: Meni će korisnicima omogućiti promenu prikaza, kao i filtriranje i sortiranje lokacija, što će poboljšati korisničko iskustvo.

Korisnički interfejs

Interfejs će biti jednostavan i pregledan, sa jasno označenim dugmadima i listama. Glavni ekran prikazuje naslov i listu kategorija, a pritiskom na kategoriju otvara se ekran sa detaljima o odgovarajućim lokacijama. Meni će biti dostupan sa gornje strane ekrana i nudiće dodatne opcije za podešavanje prikaza. Struktura ekrana može biti organizovana u dva osnovna prikaza:

- **Lista kategorija** – osnovni ekran sa izborom kategorija,
- **Detalji o lokaciji** – ekran sa informacijama o izabranoj lokaciji.

Tehnološki aspekti

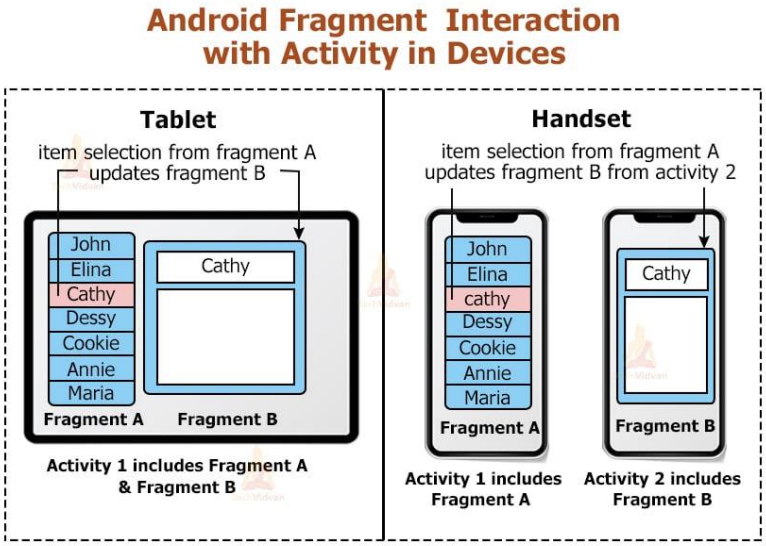
Aplikacija će se razvijati koristeći Android Studio, koristićemo Javu ili Kotlin kao programski jezik, kao i XML za kreiranje korisničkog interfejsa. S obzirom na to da je sve hardkodirano, podaci o lokacijama biće smešteni u kodu aplikacije, u vidu liste ili mape (npr. ArrayList ili HashMap), a u kasnijim verzijama ćemo ubaciti i baze podataka. U nastavku je dat primer kolekcije o lokacijama:

```
import java.util.ArrayList;
import java.util.List;

public class Lokacija {
    private String naziv;
    private String opis;
    private String kontaktTelefon;
    private String radnoVreme;

    // Konstruktor
    public Lokacija(String naziv, String opis, String telefon, String radnoVreme) {
        this.naziv = naziv;
        this.opis = opis;
        this.kontaktTelefon = telefon;
        this.radnoVreme = radnoVreme;
    }

    // Getteri i setteri
    public String getNaziv() {
        return naziv;
    }

    public String getOpis() {
        return opis;
    }

    public String getKontaktTelefon() {
        return kontaktTelefon;
    }

    public String getRadnoVreme() {
        return radnoVreme;
    }
    @Override
    public String toString() {
        return "Lokacija {" +
```

```
                "naziv ='" + naziv + '\'' +
                ", opis ='" + opis + '\'' +
                ", kontaktTelefon ='" + kontaktTelefon + '\'' +
                ", radnoVreme ='" + radnoVreme + '\'' +
                '}';
    }
}
```

Fragmenti u Androidu

Fragmenti predstavljaju samostalne delove korisničkog interfejsa koji se koriste unutar aktivnosti u Android aplikacijama. Fragmenti omogućavaju modularnost i fleksibilnost u dizajnu, jer se mogu kombinovati i menjati tokom izvršavanja aplikacije. Oni su idealni za aplikacije sa složenim rasporedom i za prilagođavanje korisničkog interfejsa različitim veličinama ekrana, kao što su tableti i pametni telefoni. Fragmenti imaju svoj životni ciklus, koji je povezan sa životnim ciklusom aktivnosti u kojoj se nalaze, ali takođe imaju određenu autonomiju.



Glavne metode u životnom ciklusu fragmenta su:

1. onAttach() – Poziva se kada se fragment pridruži aktivnosti.
2. onCreate() – Inicijalizacija fragmenta. Ova metoda je slična onCreate() metodi aktivnosti.
3. onCreateView() – Kreira i vraća pogled (View) koji će fragment prikazati.
4. onActivityCreated() – Poziva se kada je aktivnost u kojoj je fragment potpuno kreirana.
5. onStart() – Fragment postaje vidljiv korisniku.
6. onResume() – Fragment postaje interaktivan.
7. onPause() – Fragment više nije interaktivan.
8. onStop() – Fragment više nije vidljiv.
9. onDestroyView() – Briše prikaz fragmenta.
10. onDestroy() – Oslobađaju se resursi.
11. onDetach() – Fragment se odvaja od aktivnosti.

Fragmenti često komuniciraju sa aktivnostima kako bi podelili neke podatke. To se postiže implementiranjem interfejsa u fragmentu koji aktivnost mora da implementira. Sledi primer koji prikazuje kako se koristi fragment u Android aplikaciji. On uključuje kreiranje jednostavnog fragmenta koji prikazuje tekst i dugme. Kada korisnik klikne na dugme, prikazuje se **Toast** poruka. Takođe, prikazano je kako da se fragment doda u aktivnost.

Ako želite da implementirate i isprobate ovo, prvo kreirajte novi XML fajl unutar foldera res/layout sa nazivom **fragment_example.xml** i unesite sledeći kod:

```
<?xml version="1.0" encoding="utf-8"?>
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:gravity="center">
    <TextView
        android:id="@+id/textView"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Ovo je primer fragmenta"
        android:textSize="18sp"
        android:textColor="#000000"/>
    <Button
        android:id="@+id/button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Klikni me"
        android:layout_marginTop="16dp"/>
</LinearLayout>
```

Ovaj XML fajl definiše LinearLayout sa TextView i Button koji će se prikazati u fragmentu.

Zatim, treba kreirati novu Java klasu za fragment. Ovaj fragment će naslediti **Fragment** klasu i sadržaće metode za inicijalizaciju prikaza.

```
import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.Button;
import android.widget.Toast;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;
```

```
public class ExampleFragment extends Fragment {
    // Metoda koja kreira i inicijalizuje izgled fragmenta
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
                             @Nullable ViewGroup container,
                             @Nullable Bundle savedInstanceState) {
        // Inflater objekat koristi XML fajl za definisanje prikaza fragmenta.
        // 'inflater.inflate' metoda kreira View objekat na osnovu XML-a fragment_example
        View view = inflater.inflate(R.layout.fragment_example, container, false);
        // Pronalazimo dugme koje smo definisali u XML-u pomoću ID-a `button`
        Button button = view.findViewById(R.id.button);
        // Postavljamo OnClickListener na dugme kako bismo reagovali na korisnikov klik
        button.setOnClickListener(new View.OnClickListener() {
            @Override
            public void onClick(View v) {
                // Kada korisnik klikne na dugme, prikazaće se Toast poruka
                Toast.makeText(getActivity(), "Klik klik!", Toast.LENGTH_SHORT).show();
            }
        });
        // Vraćamo View objekat kako bi fragment mogao da se prikaže
        return view;
    }
}
```

Sada ćemo dodati fragment u glavnu aktivnost aplikacije. Prvo dodajemo kod u activity_main.xml a zatim i kod u MainActivity.java. Oba koraka su potrebna da bi se fragment uspešno prikazao.

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

FrameLayout služi kao kontejner u kojem će fragment biti prikazan.

Konačno, prelazimo na MainActivity.java fajl i dodajmo sledeći kod u onCreate metodi:

```
import android.os.Bundle;
import androidx.appcompat.app.AppCompatActivity;
import androidx.fragment.app.FragmentManager;
import androidx.fragment.app.FragmentTransaction;
```

```
public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        // Proverite da li se fragment već nalazi u kontejneru
        if (savedInstanceState == null) {
            // Kreirajte instancu fragmenta
            ExampleFragment exampleFragment = new ExampleFragment();
            // Pozovite FragmentManager i započnite FragmentTransaction
            FragmentManager fragmentManager = getSupportFragmentManager();
            FragmentTransaction fragmentTransaction = fragmentManager.beginTransaction();
            // Dodajte fragment u kontejner
            fragmentTransaction.add(R.id.fragment_container, exampleFragment);
            fragmentTransaction.commit();
        }
    }
}
```

Ovaj kod koristi **FragmentManager** i **FragmentTransaction** kako bi dodao ExampleFragment u FrameLayout koji smo kreirali u activity_main.xml fajlu. Proveravamo da li je savedInstanceState null da bismo izbegli ponovno dodavanje fragmenta pri rotaciji ekrana ili obnovi aktivnosti.

Kada pokrenete aplikaciju, u MainActivity će se prikazati **ExampleFragment** sa tekstom i dugmetom. Kada korisnik klikne na dugme, prikazaće se Toast poruka sa tekstom **"Klik klik!".**

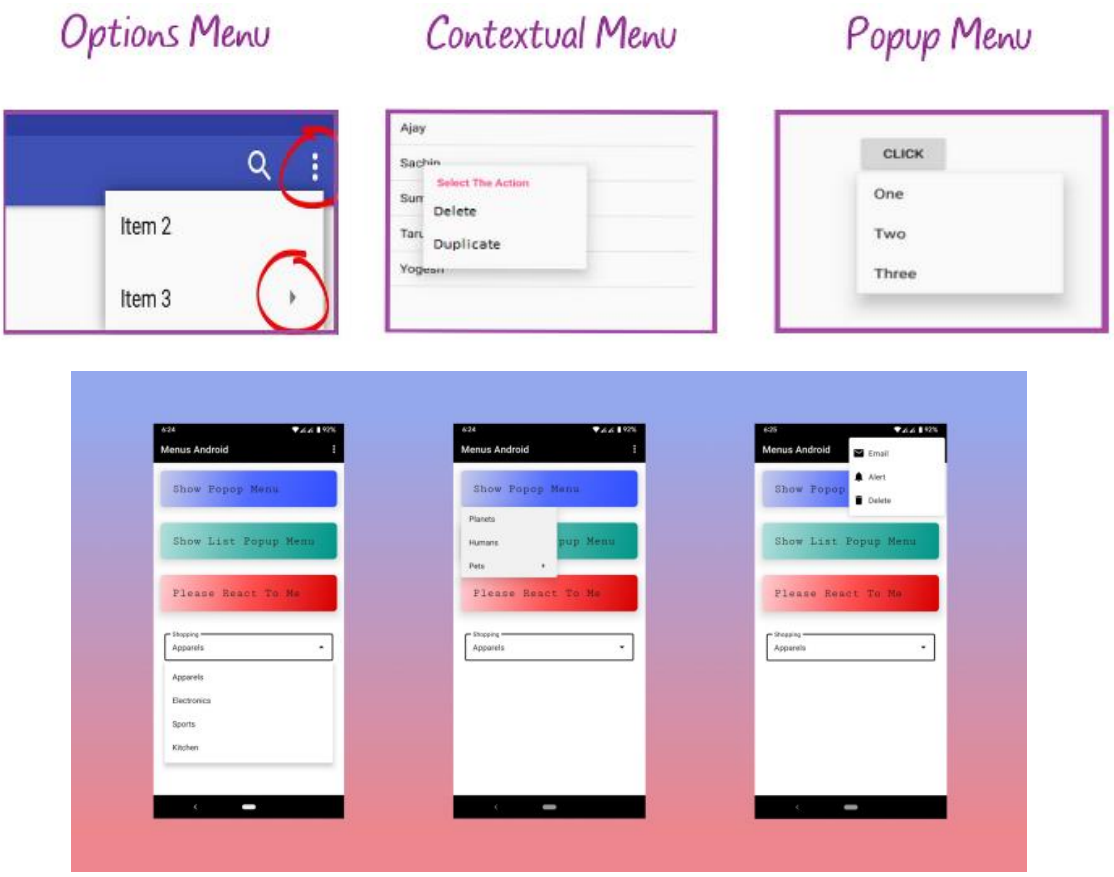
Kao što možete videti, fragmenti su korisni za kreiranje više panela u aplikaciji, gde na većim uređajima kao što su tableti mogu da prikažu dva panela jedan pored drugog, dok na manjim ekranima mogu da se prikažu jedan po jedan. Fragmenti omogućavaju prilagođavanje interfejsa na osnovu datog uređaja i unapređuju korisničko iskustvo.

Meniji (Menus) u Androidu

Meni (Menu) u Android aplikacijama omogućava korisnicima da pristupe različitim funkcijama i opcijama unutar aplikacije. Meniji u Androidu se smatraju ključnim delom korisničkog interfejsa, jer omogućavaju jednostavan pristup funkcionalnostima bez zauzimanja prostora na ekranu. Postoje različite vrste menija u Androidu:

- **Options Menu** (Opcioni meni): Ovo je osnovni meni koji se obično nalazi u zaglavlju aplikacije i pojavi se kao tri tačkice u gornjem desnom uglu ekrana. Ovaj meni pruža osnovne opcije kao što su postavke, informacije o aplikaciji ili opcije za deljenje sadržaja.
- **Context Menu** (Kontekstualni meni): Ovaj meni se pojavljuje kada korisnik zadrži stisak na određeni element u aplikaciji. Najčešće se koristi za opcije specifične za određeni element, kao što su brisanje, uređivanje ili kopiranje.
- **Popup Menu** (Iskačući meni): Ovaj meni se prikazuje kao plutajući meni na ekranu i obično je povezan sa određenim dugmetom ili akcijom. Popup meni se koristi kada su potrebne brze opcije za određene akcije.

Da bi se kreirao meni, Android koristi XML datoteke koje se nalaze u res/menu direktorijumu. U XML fajlu definišemo stavke menija, a zatim u kodu pristupamo tim stavkama i postavljamo akcije za njih. Ovo je primer jednog opcionog menija, koji je realizovan preko fajla **menu_main.xml.**




```
<!-- res/menu/menu_main.xml -->
<?xml version="1.0" encoding="utf-8"?>
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/action_settings"
        android:title="Settings"
        android:orderInCategory="100"
        android:showAsAction="never" />
    <item
        android:id="@+id/action_about"
        android:title="About"
        android:orderInCategory="101"
        android:showAsAction="never" />
</menu>
```

<menu> je glavni element koji sadrži stavke menija , a **<item>** predstavlja pojedinačnu stavku u meniju. Svaka stavka može imati ID (android:id), naziv (android:title), redosled u kategoriji (android:orderInCategory), i showAsAction, što definiše da li će se stavka prikazivati kao ikona ili u padajućem meniju.

U glavnoj aktivnosti, meniji se prikazuju korišćenjem metode onCreateOptionsMenu, gde se integriše (inflate operacija) meni XML fajl i prikazuje u zaglavlju aplikacije. Sledi primer koda:

```
import android.os.Bundle;
import android.view.Menu;
import android.view.MenuInflater;
import android.view.MenuItem;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;

public class MainActivity extends AppCompatActivity {
    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
    }
    @Override
    public boolean onCreateOptionsMenu(Menu menu) {
        // Inflate menija koristi XML fajl za kreiranje stavki menija
        MenuInflater inflater = getMenuInflater();
        inflater.inflate(R.menu.menu_main, menu);
        return true;
    }
    @Override
    public boolean onOptionsItemSelected(MenuItem item) {
        // Upravljanje klikovima na stavke menija
        switch (item.getItemId()) {
            case R.id.action_settings:
                // Prikazivanje poruke kada je kliknuto na "Settings"
                Toast.makeText(this, "Settings selected", Toast.LENGTH_SHORT).show();
                return true;
            case R.id.action_about:
                // Prikazivanje poruke kada je kliknuto na "About"
                Toast.makeText(this, "About selected", Toast.LENGTH_SHORT).show();
                return true;
            default:
                return super.onOptionsItemSelected(item);
        }
    }
}
```

Kada pokrenete aplikaciju na vrhu ekrana videćete ikonu menija sa tri tačkice. Kada kliknete na ovu ikonu, pojaviće se stavke menija definisane u menu_main.xml, poput "Settings" i "About". Kada kliknete na neku stavku menija, prikazuje se odgovarajuća Toast poruka.

Događaji (Events) u Androidu

Događaji su akcije koje korisnici preduzimaju na uređaju, kao što su dodiri, klikanja, pomeranja i slično. U Androidu, svi ovi korisnički interfejs elementi mogu slati obaveštenja (events) aplikaciji o tome što se dogodilo, što omogućava aplikaciji da reaguje na korisničke akcije.

Postoji nekoliko vrsta događaja koje Android može da obradi:

- **Događaji Dodira (Touch Events)**: Ovi događaji se javljaju kada korisnik dodirne ili pomera prst po ekranu. Ključne klase uključuju MotionEvent, koji predstavlja pojedinačne dodirne interakcije.
- **Događaji Klika (Click Events)**: Javljaju se kada korisnik klikne na dugme ili neki drugi UI element. Ovi događaji se obično koriste za pokretanje akcija u aplikaciji.
- **Događaji Tastature (Keyboard Events)**: Javljaju se kada korisnik koristi fizičku ili virtuelnu tastaturu. Ključne klase uključuju KeyEvent.
- **Događaji Pomeranja (Scroll Events)**: Ovi događaji se koriste kada korisnik pomera sadržaj unutar scrollable view-a (npr. ScrollView).
- **Događaji Fokusa (Focus Events)**: Ovi događaji se javljaju kada korisnik prebacuje fokus između različitih UI elemenata.

Da bi aplikacija reagovala na događaje, morate da registrujete slušaoce (listeners). Slušaoci su objekti koji implementiraju određene interfejse koji definišu kako aplikacija treba da reaguje na događaje. Sledi jednostavan primer koda:

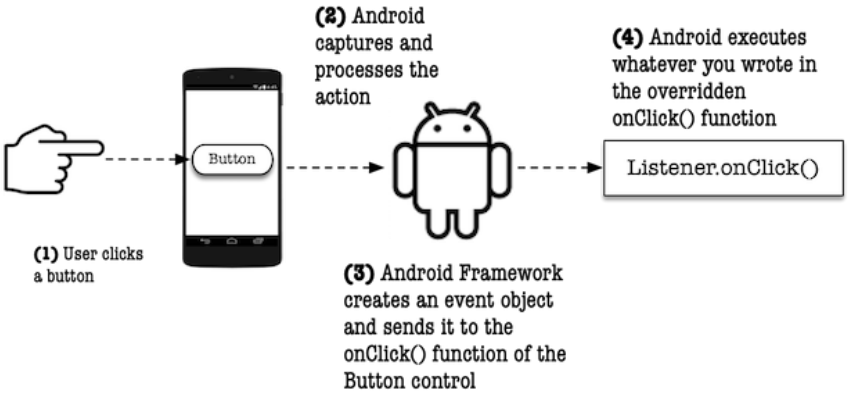
```
Button myButton = findViewById(R.id.my_button);

myButton.setOnClickListener(new View.OnClickListener() {

    @Override

    public void onClick(View v) {

        // Akcija koja će se izvršiti kada korisnik klikne na dugme

        Toast.makeText(getApplicationContext(), "Klik klik!", Toast.LENGTH_SHORT).show();

    }

});
```

Obrada događaja se može vršiti na nekoliko načina:

- **Direktno u Aktivnosti**: Možete dodati slušaoce događaja direktno u aktivnost. Ovo je jednostavno, ali može postati neuredno ako aktivnost postane prevelika.
- **Korišćenjem Klasa ili Fragmenta**: Uklanjanje logike iz aktivnosti može znatno poboljšati organizaciju koda. Kreiranjem zasebnih klasa ili fragmenta za obradu događaja možete imati bolje odvajanje logike.

U nekim slučajevima, možda će vam biti potrebna složenija logika za obradu događaja, kao što su gestovi (gestures) ili višestruki dodiri (multi-touch). Android pruža različite alate i klase za prepoznavanje gestova, kao što su **GestureDetector i ScaleGestureDetector**. U nastavku je dat jedan bazični primer GestureDetectora za dvostruki dodir na ekranu:

```
GestureDetector gestureDetector = new GestureDetector(this, new
GestureDetector.SimpleOnGestureListener() {

    @Override

    public boolean onDoubleTap(MotionEvent e) {

        // Akcija koja se izvršava na dvostruki dodir

        return true;

    }

});

myView.setOnTouchListener(new View.OnTouchListener() {

    @Override

    public boolean onTouch(View v, MotionEvent event) {

        gestureDetector.onTouchEvent(event);

        return true;

    }

});
```



U nastavku je još jedan primer kako možete kombinovati EditText polje i događaj kada korisnik pritisne Enter taster. Ovo može biti korisno za više aplikacija.

```
EditText editText = findViewById(R.id.my_edit_text);

editText.setOnKeyListener(new View.OnKeyListener() {

    @Override

    public boolean onKey(View v, int keyCode, KeyEvent event) {

      if (event.getAction() == KeyEvent.ACTION_DOWN && keyCode == KeyEvent.KEYCODE_ENTER) {

            // Akcija kada korisnik pritisne "Enter" dok je fokus na EditText

               String inputText = editText.getText().toString();

               Toast.makeText(getApplicationContext(), "Unenesen je tekst: " + inputText,
               Toast.LENGTH_SHORT).show();

               return true; // Oznaka da je događaj obrađen

        }

        return false; // Oznaka da događaj nije obrađen

    }

});
```

Koraci za izradu aplikacije

Korak 1: Priprema okruženja

- **Kreiranje novog projekta**:
  - Otvorite Android Studio i izaberite "New Project".
  - Izaberite "Empty Activity" kao šablon.
  - Unesite naziv projekta (npr. TouristInfoApp) i odaberite Javu kao programski jezik.
  - Kliknite na "Finish" da biste završili kreiranje projekta.

Korak 2: Kreiranje modela podataka

- **Definisanje klase Lokacija**:
  - Napravite novu Java klasu pod imenom Lokacija unutar paketa model.
  - Dodajte kod koji definiše klasu Lokacija (sa atributima kao što su naziv, opis, kontakt telefon, radno vreme i odgovarajući getteri i setteri).

Korak 3: Dizajn korisničkog interfejsa

Otvorite **activity_main.xml** u res/layout folderu i dodajte ListView za prikaz kategorija, kao i ostale neophodne komponente. Sledi primer dizajna:

```
<LinearLayout

    xmlns:android="http://schemas.android.com/apk/res/android"

    android:layout_width="match_parent"

    android:layout_height="match_parent"

    android:orientation="vertical"

    android:padding="16dp">



    <!-- Naslov -->

    <TextView

        android:id="@+id/locationTitle"

        android:layout_width="wrap_content"

        android:layout_height="wrap_content"

        android:text="Izabrana Lokacija"

        android:textSize="24sp"

        android:textStyle="bold"

        android:layout_marginBottom="16dp"/>



    <!-- Prikaz lokacije -->

    <TextView
```

```
        android:id="@+id/locationInfo"
        android:layout_width="match_parent"
        android:layout_height="wrap_content"
        android:text="Detalji o lokaciji će biti ovde prikazani"
        android:textSize="16sp"
        android:layout_marginBottom="16dp"/>
    <!-- Dugme za prikaz dodatnih informacija -->
    <Button
        android:id="@+id/showDetailsButton"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:text="Prikaži Detalje"
        android:layout_gravity="center"/>
</LinearLayout>
```

Korak 4: Implementacija Fragmenta

- **Kreiranje fragmenta za detalje o lokaciji**:
  - Napravite novi fragment pod imenom DetailFragment.
  - U res/layout, kreirajte XML fajl pod nazivom **fragment_detail.xml** i definišite layout sa TextView za prikaz detalja o lokaciji.

Primer dizajna i logike je dat u nastavku:

```
<LinearLayout xmlns:android="http://schemas.android.com/apk/res/android"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    android:orientation="vertical"
    android:padding="16dp">

    <TextView
        android:id="@+id/textViewNaziv"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:textSize="20sp"/>

    <TextView
        android:id="@+id/textViewOpis"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
```

```
    <TextView
        android:id="@+id/textViewKontakt"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>

    <TextView
        android:id="@+id/textViewRadnoVreme"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"/>
</LinearLayout>
```

ARG_LOKACIJA je ključno ime koje se koristi za prebacivanje objekta Lokacija između fragmenta i aktivnosti. Metoda *newInstance* omogućava kreiranje nove instance fragmenta s prosleđenim *Lokacija* objektom, koji se smešta u *Bundle* radi lakšeg prenosa podataka između fragmenta i aktivnosti. Kada se fragment kreira, poziva se metoda *onCreateView*, koja uzima dati layout (R.layout.fragment_detail) i inicijalizuje TextView elemente za prikaz informacija o lokaciji. Da bismo dobili prosleđene podatke, klasa koristi getArguments() metodu, a ako lokacija nije null, odgovarajući podaci se postavljaju u TextView (naziv, opis, kontakt telefon i radno vreme). Na kraju, metoda vraća *View* komponentu, koja predstavlja korisnički interfejs fragmenta.

```
package com.example.touristinfoapp;

import android.os.Bundle;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.TextView;
import androidx.annotation.NonNull;
import androidx.annotation.Nullable;
import androidx.fragment.app.Fragment;

public class DetailFragment extends Fragment {

    private static final String ARG_LOKACIJA = "lokacija";

    public static DetailFragment newInstance(Lokacija lokacija) {
        DetailFragment fragment = new DetailFragment();
        Bundle args = new Bundle();
        args.putSerializable(ARG_LOKACIJA, lokacija);
        fragment.setArguments(args);
        return fragment;
    }
```

```
    @Nullable
    @Override
    public View onCreateView(@NonNull LayoutInflater inflater,
                @Nullable ViewGroup container, @Nullable Bundle savedInstanceState) {
        View view = inflater.inflate(R.layout.fragment_detail, container, false);
        TextView textViewNaziv = view.findViewById(R.id.textViewNaziv);
        TextView textViewOpis = view.findViewById(R.id.textViewOpis);
        TextView textViewKontakt = view.findViewById(R.id.textViewKontakt);
        TextView textViewRadnoVreme = view.findViewById(R.id.textViewRadnoVreme);

        Lokacija lokacija = (Lokacija) getArguments().getSerializable(ARG_LOKACIJA);
        if (lokacija != null) {
            textViewNaziv.setText(lokacija.getNaziv());
            textViewOpis.setText(lokacija.getOpis());
            textViewKontakt.setText(lokacija.getKontaktTelefon());
            textViewRadnoVreme.setText(lokacija.getRadnoVreme());
        }
        return view;
    }
}
```

Korak 5: Implementacija Menija

- **Dodavanje Menija**
  - Kreirajte XML fajl **menu_main.xml** u res/menu folderu i dodajte opcije (npr. sortiranje, filtriranje).

```
<menu xmlns:android="http://schemas.android.com/apk/res/android">
    <item
        android:id="@+id/action_sort"
        android:title="Sortiraj"/>
    <item
        android:id="@+id/action_filter"
        android:title="Filtriraj"/>
</menu>
```

- **Prikaz Menija u aktivnosti**
  - U MainActivity.java, treba implementirati metodu za integrciju Menija:

```
@Override
public boolean onCreateOptionsMenu(Menu menu) {
    MenuInflater inflater = getMenuInflater();
    inflater.inflate(R.menu.menu_main, menu);
    return true;
}
```

Korak 6: Dodavanje funkcionalnosti

- **Implementacija logike za izbor kategorije**
  - Da biste uzeli odabranu lokaciju iz liste kada korisnik klikne na dugme, prvo je potrebno da postavite OnItemClickListener za ListView, kako biste sačuvali izabranu lokaciju. Zatim, u OnClickListener za dugme, jednostavno koristite tu sačuvanu lokaciju. Sve se ovo odvija u MainActivity.java fajlu.

- **Popunjavanje liste kategorija**
  - U MainActivity.java, dodajte kod za popunjavanje liste kategorija koristeći **ArrayAdapter** sa listom instanci Lokacija. Adapter je ključni deo Android aplikacija koji povezuje podatke (kao što su lista lokacija) sa vizuelnim komponentama (poput ListView). Sledi jedan primer adaptera:

```
import android.content.Context;
import android.view.LayoutInflater;
import android.view.View;
import android.view.ViewGroup;
import android.widget.ArrayAdapter;
import android.widget.TextView;
import java.util.List;

// Klasa LokacijaAdapter koja nasleđuje ArrayAdapter za klasu Lokacija
public class LokacijaAdapter extends ArrayAdapter<Lokacija> {

    // Konstruktor za adapter koji prima kontekst i listu lokacija
    public LokacijaAdapter(Context context, List<Lokacija> lokacije) {
        super(context, 0, lokacije); // Pozivamo super konstruktor sa praznim layout-om
    }

    // Metoda koja se poziva za svaki element u listi
    @Override
    public View getView(int position, View convertView, ViewGroup parent) {
        // Uzmi trenutnu lokaciju na osnovu pozicije
        Lokacija lokacija = getItem(position);

        // Proveri da li se koristi postojeći prikaz, inače daj novi prikaz
        if (convertView == null) {
            // Uzimanje novog layout-a za prikaz lokacije
            convertView = LayoutInflater.from(getContext()).inflate(R.layout.item_lokacija,
                                                                    parent, false);
        }
```

```
        // Pronađi TextView u layout-u za naziv lokacije
        TextView textViewNaziv = convertView.findViewById(R.id.textViewNaziv);
        // Pronađi TextView u layout-u za opis lokacije
        TextView textViewOpis = convertView.findViewById(R.id.textViewOpis);

        // Postavi naziv lokacije u TextView
        textViewNaziv.setText(lokacija.getNaziv());
        // Postavi opis lokacije u TextView
        textViewOpis.setText(lokacija.getOpis());

        // Vrati prikaz koji se može renderovati na ekranu
        return convertView;
    }
}
```

Kada kreirate **LokacijaAdapter**, dodajte ga u **ListView**. Sada je dat primer MainActivity-ja kako sve ovo može da izgleda:

```
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.ListView;
import android.widget.Toast;
import androidx.appcompat.app.AppCompatActivity;
import java.util.ArrayList;
import java.util.List;

public class MainActivity extends AppCompatActivity {
    private Lokacija odabranaLokacija; // Promenljiva za odabranu lokaciju
    private LokacijaAdapter adapter; // Adapter za listu lokacija

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        setContentView(R.layout.activity_main);
        ListView listViewCategories = findViewById(R.id.listViewCategories);

        // Kreiranje liste lokacija
        List<Lokacija> lokacije = new ArrayList<>();
```

```
        // Dodavanje nekoliko lokacija u listu
        lokacije.add(new Lokacija("Beograd", "Glavni grad Srbije", "011-123456"));
        lokacije.add(new Lokacija("Novi Sad", "Grad kulture", "021-654321"));
        lokacije.add(new Lokacija("Niš", "Grad sa bogatom istorijom", "018-777888"));

        // Postavite adapter za ListView
        adapter = new LokacijaAdapter(this, lokacije);
        listViewCategories.setAdapter(adapter);

        // Postavite OnItemClickListener za ListView
        listViewCategories.setOnItemClickListener((parent, view, position, id) -> {
            odabranaLokacija = (Lokacija) parent.getItemAtPosition(position);
        });

        // Postavite OnClickListener za dugme
        Button showDetailsButton = findViewById(R.id.showDetailsButton);
        showDetailsButton.setOnClickListener(v -> {
            if (odabranaLokacija != null) {
                DetailFragment detailFragment =
                 DetailFragment.newInstance(odabranaLokacija);
                 getSupportFragmentManager().beginTransaction()
                    .replace(R.id.fragment_container, detailFragment)
                    .addToBackStack(null)
                    .commit();
            } else {
                // Opcionalno: Prikazati obaveštenje korisniku da izabere lokaciju
                Toast.makeText(MainActivity.this, "Molimo izaberite lokaciju!",
                                                    Toast.LENGTH_SHORT).show();
            }
        });
    }
}
```

Korak 7: Testiranje aplikacije

- **Pokretanje aplikacije**
  - Pokrenite emulator ili povežite Android uređaj i pokrenite aplikaciju
  - Proverite funkcionalnost klikova, prikaz menija i navigaciju između ekrana.

Korisni resursi

Odličan tutorijal za kreiranje fragmenta:
https://www.youtube.com/watch?app=desktop&v=cto1algelZA

Jednostavan tutorijal za kreiranje menija:
https://www.youtube.com/watch?v=5l6oDbE4zaM

Malo napredniji meni, sa dinamičkim elementima, obavezno pogledati:
https://www.youtube.com/watch?v=Es5UFII4oak

Jednostavan tutorijal za eventove:
https://www.youtube.com/watch?v=LV_5lOvYAn8

Malo naporniji Indijac, ali lepo objašnjava u tutorijalu:
https://www.youtube.com/watch?v=uY9iZiamyZs

Zvanična dokumentacija:

https://developer.android.com/guide/fragments/create

https://developer.android.com/develop/ui/views/components/menus

https://developer.android.com/develop/ui/views/touch-and-input/input-events

https://developer.android.com/topic/architecture/ui-layer/events

https://www.tutorialspoint.com/android/android_event_handling.htm

Još korisnih sadržaja:

https://abhiandroid.com/ui/fragment#gsc.tab=0

https://medium.com/@androidcookies1/menu-types-in-android-with-tutorial-4ebd402ebf83

https://medium.com/@zorbeytorunoglu/event-patterns-in-android-kotlin-bcfdd254e44b