АКАДЕМИЈА ТЕХНИЧКО-ВАСПИТАЧКИХ СТРУКОВНИХ СТУДИЈА
ОДСЕК НИШ

ANDROID PROGRAMIRANJE

Vežba 5: Razvoj *NearMe* aplikacije

Teorijske napomene

1. Google Maps API

Google Maps API je alat koji omogućava programerima da implementiraju interaktivne mape u mobilne aplikacije. Pomoću ovog API-ja možete da prikazujete mape, dodate markere na mapu, crtate rute, i pružite korisnicima informacije o geografskim lokacijama. Google Maps API je ključni alat za mobilne aplikacije koje se bave lociranjem korisnika, navigacijom i funkcionalnostima vezanim za geografske podatke. U ovom projektu ćemo koristiti Google Maps API da prikažemo mapu na kojoj se može pratiti trenutna lokacija korisnika i označiti određene vrste mesta u okolini.

Osnovne funkcionalnosti Google Maps API-ja:

- **Mape**: Prikazivanje mapa sa raznim slojevima.
- **Markeri**: Označavanje tačaka na mapi sa opisima.
- **Geolokacija**: Prikazivanje trenutne lokacije korisnika na mapi.
- **Route**: Prikazivanje rute između dve tačke.

2. Google Places API

Google Places API omogućava aplikacijama da pretražuju i koriste informacije o mestima, kao što su restorani, tržni centri, teretane, fakulteti, muzeji i drugi objekti. Places API omogućava da pretražujete mesta u određenom radijusu od korisničke lokacije, kao i da prikazujete detalje o tim mestima, uključujući adresu, tip i druge relevantne informacije.

Glavne komponente Google Places API-ja:

- **Place Search**: Pretraga mesta po tipu i lokaciji.
- **Place Details**: Detaljne informacije o određenom mestu (adresa, telefon, radno vreme).
- **Place Photos**: Prikazivanje slika mesta.
- **Place Autocomplete**: Automatsko popunjavanje pretrage na osnovu delimično unetih podataka.

Za potrebe ove vežbe koristićemo **Place Search** da bismo pretraživali mesta kao što su restorani, tržni centri i teretane u okolini korisničke lokacije.

3. Location Services

Android Location Services omogućavaju aplikacijama da prepoznaju i koriste geografsku lokaciju korisnika. Korišćenjem Location Services, aplikacije mogu dobiti tačne ili približne informacije o trenutnoj lokaciji korisnika, bilo putem GPS-a, Wi-Fi mreže ili mobilnih mreža. Android pruža nekoliko API-ja za rad sa lokacijama, ali **FusedLocationProviderClient** je najpouzdaniji i najefikasniji način za pristup lokaciji, jer automatski bira najbolji izvor podataka (GPS, Wi-Fi, mreža) u zavisnosti od uslova.

Vrste lokacija:

- **Precizna lokacija (GPS)**: Obezbeđuje visoku tačnost, ali može biti zahtevna za bateriju.
- **Približna lokacija (Wi-Fi, mobilna mreža)**: Manje precizna, ali koristi manje resursa i može raditi čak i kada GPS signal nije dostupan.

Detalji implementacije

Kroz ovu vežbu razvijamo aplikaciju koja korisnicima omogućava da vide svoju trenutnu lokaciju na mapi, a zatim iz padajuće liste odaberu vrstu mesta (restoran, tržni centar, teretana) i pritisnu dugme kako bi na mapi prikazali odabrane objekte. Korišćenje Google Places API-ja biće ključno za pribavljanje relevantnih lokacija na mapi.

1. Kreiranje mape i postavljanje na ekranu

Prvi korak u implementaciji je postavljanje mape na ekranu. Kada korisnik pokrene aplikaciju, mapa se prikazuje, a korisnička lokacija se automatski pronalazi i dodaje na ekranu. Mapa koristi GoogleMap objekat, koji omogućava prikazivanje i manipulaciju mapom. Da bismo ovo postigli:

- Koristićemo **SupportMapFragment** u XML layout-u kako bismo integrisali mapu.
- Kada se mapa učita, koristićemo **FusedLocationProviderClient** da dobijemo trenutnu lokaciju korisnika i postavimo kameru na tu tačku.

2. Pretraga Mesta u okolini

Aplikacija omogućava korisnicima da odaberu vrstu mesta (restoran, tržni centar, teretana) i da vide koja mesta se nalaze u njihovoj blizini. Da bismo ovo postigli, koristićemo **Google Places API**. Kada korisnik odabere tip mesta iz spinner-a i pritisne dugme, koristićemo Places API da pretražimo mesta u okolini i dobijemo informacije o tim mestima (naziv, tip, lokacija). Onda ćemo koristiti **MarkerOptions** da dodamo markere na mapu.

3. Prikazivanje mesta na mapi

Kada pretražimo mesta u okolini, rezultati se prikazuju na mapi u obliku markera. Markeri označavaju tačne lokacije mesta koja korisnik traži (restorane, teretane, tržne centre). Markeri se dodaju pomoću **mMap.addMarker()** i mogu sadržati neke osnovne informacije o mestu.

4. Interakcija sa Mapom

Korisnik može da klikne na marker na mapi i dobije više informacija o tom mestu, kao što su naziv, adresa i možda slike. Ovo može biti realizovano korišćenjem **Marker** objekta i implementacijom **OnMarkerClickListener**.

5. Dinamičko učitavanje mesta

Aplikacija treba da učita rezultate sa servera (Places API) dinamički, zavisno od korisničkog odabira. Kada korisnik promeni odabrani tip mesta u spinner-u, dugme pokreće novu pretragu i mapu ažurira sa novim markerima za izabrani tip mesta.

6. Optimizacija i upravljenje lokacijama

Kao što je već naglašeno, preporučuje se korišćenje **FusedLocationProviderClient** za brzo i efikasno dobijanje korisničke lokacije. Takođe, treba pratiti dozvole za pristup lokaciji, poput **ACCESS_FINE_LOCATION** i **ACCESS_COARSE_LOCATION** u AndroidManifest.xml, kao i u kodu na runtime-u, kako bi aplikacija bila u skladu sa bezbednosnim pravilima.

Koraci za izradu aplikacije

1. Kreiranje Novog Projekta

- Otvorite Android Studio i kreirajte novi projekat. Odaberite **Empty Activity**.
- Nazovite projekat **NearMe.**
- Postavite minimalnu verziju aplikacije na **API 21** ili višu.

2. Dodavanje Zavisnosti u build.gradle

Morate dodati sledeće zavisnosti u build.gradle kako biste koristili Google Maps i Places API.

U build.gradle (Module: app), dodajte ove zavisnosti:

```
dependencies {
    implementation 'com.google.android.gms:play-services-maps:17.0.1'
    implementation 'com.google.android.gms:play-services-location:17.0.0'
    implementation 'com.google.android.libraries.places:places:2.7.0'
}
```

U build.gradle (Project: NearMe), dodajte:

```
allprojects {
    repositories {
        google()  // Za korišćenje Google servisa
        jcenter()
    }
}
```

3. Generisanje API Ključa

Idite na https://console.cloud.google.com/welcome/new?pli=1 i:

- Prijavite se sa Google nalogom: Ako već nemate nalog, potrebno je da se registrujete na Google Cloud.

- Kliknite na "Select a Project".
- Zatim kliknite na "New Project".
- Unesite ime projekta, odaberite Location i kliknite na Create dugme.
- Sada u pretrazi Google Cloud konzole upišite *"Maps SDK for Android"* i kliknite na rezultat.
- Kliknite na Enable dugme kako biste omogućili Maps SDK for Android za vaš projekat.
- Taj process ponovite i za *“Places Api”*.
- Nakon toga, potrebno je generisati API ključ, koji ćete koristiti u Android aplikaciji za pristup Google Mapama i Places API-ju.
- U Google Cloud konzoli idite na APIs & Services > Credentials u meniju sa leve strane.
- Kliknite na "Create Credentials" (gore na stranici) i izaberite "API Key". API ključ će biti generisan na taj način.
- Posle toga, automatski će biti prikazan vaš novi API ključ. Možete ga kopirati za kasnije.
- Opcionalno, da biste povećali sigurnost, možete ograničiti API ključ na određene uslove. To možete uraditi klikom na "Restrict Key":
    - HTTP referrers: Ovaj parametar možete postaviti ako želite da API ključ bude dostupan samo sa određenih URL-ova (npr. samo sa vaše aplikacije).
    - API restrictions: Ovde možete ograničiti ključ samo na specifične API-je kao što su Maps SDK for Android ili Places API. Preporučuje se da ograničite ključ na ove API-je kako biste sprečili neovlašćeno korišćenje.
- Kada dobijete API ključ, treba ga dodati u AndroidManifest.xml (unutar <application> taga), da bli aplikacija mogla da koristi Google Maps I Places API.

```
<meta-data
        android:name="com.google.android.maps.v2.API_KEY"
        android:value="@string/google_maps_key" />
```

- Onda dodajte ključ u res/values/strings.xml:

```
<resources>
        <string name="google_maps_key">YOUR_API_KEY_HERE</string>
</resources>
```

Bezbednosne napomene:

Preporučuje se da uvek ograničite svoj API ključ na specifične uslove. Možete ograničiti ključ da se koristi samo sa određenim **API-ima** (kao što su Google Maps SDK ili Places API), čime sprečavate upotrebu ključa za neovlašćene API-je. Na taj način, čak i ako neko pokuša da koristi vaš ključ za pristup drugim Google API-ima, neće moći da ih koristi.

Takođe, nikada ne otkrivajte svoj API ključ. Uvek se trudite da API ključevi ne budu javni. Na primer, ne postavljajte ih u javne repozitorijume.

4. Postavljanje Layout-a

Kreirajte layout za vašu aktivnost u res/layout/activity_main.xml koji će sadržavati **MapFragment** za prikaz mape i **Spinner** za odabir tipa mesta (restoran, tržni centar, teretana):

```
<?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">

    <com.google.android.gms.maps.SupportMapFragment
        android:id="@+id/mapFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent" />

    <Spinner
        android:id="@+id/places_spinner"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_alignParentTop="true"
        android:layout_centerHorizontal="true"
        android:layout_marginTop="20dp" />

    <Button
        android:id="@+id/search_button"
        android:layout_width="wrap_content"
        android:layout_height="wrap_content"
        android:layout_below="@id/places_spinner"
        android:layout_centerHorizontal="true"
        android:text="Search" />
</RelativeLayout>
```

Ovo je samo jedan predlog dizajna, svakako možete ubaciti komponente po želji. Obavezno je da se u aplikaciji nađe jedna slika, bilo kao baner, bilo kao logo. Boje treba da budu svetle i pregledne.

5. Postavljanje MainActivity

U MainActivity.java, inicijalizujemo mapu, pretragu i korisničku lokaciju. Kada korisnik odabere vrstu mesta i pritisne dugme, pretražiće se odgovarajući tipovi mesta pomoću Places API-ja.

```
import android.os.Bundle;
import android.view.View;
import android.widget.Button;
import android.widget.Spinner;
import android.widget.ArrayAdapter;
import android.widget.Toast;

import androidx.fragment.app.FragmentActivity;

import com.google.android.gms.maps.CameraUpdateFactory;
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.model.LatLng;
import com.google.android.gms.maps.model.MarkerOptions;
import com.google.android.gms.location.FusedLocationProviderClient;
import com.google.android.gms.location.LocationServices;
import com.google.android.gms.tasks.OnSuccessListener;
import com.google.android.gms.tasks.Task;
import com.google.android.libraries.places.api.Places;
import com.google.android.libraries.places.api.model.Place;
import com.google.android.libraries.places.api.model.PlacesStatusCodes;
import com.google.android.libraries.places.api.net.FindCurrentPlaceRequest;
import com.google.android.libraries.places.api.net.PlacesClient;

import java.util.Arrays;
import java.util.List;

public class MainActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;
    private FusedLocationProviderClient fusedLocationClient;
    private Spinner placesSpinner;
    private Button searchButton;
    private PlacesClient placesClient;
```

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

    // Inicijalizujte Places API
    if (!Places.isInitialized()) {
        Places.initialize(getApplicationContext(),
                          getString(R.string.google_maps_key));
    }
    placesClient = Places.createClient(this);

    // Pronađite mapu
    SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.mapFragment);
    mapFragment.getMapAsync(this);

    // Inicijalizujte UI komponente
    placesSpinner = findViewById(R.id.places_spinner);
    searchButton = findViewById(R.id.search_button);

    // Postavite adapter za Spinner
    ArrayAdapter<String> adapter=new ArrayAdapter<>(this,
                                android.R.layout.simple_spinner_item,
                                Arrays.asList("Restaurant", "Shopping Mall", "Gym"));
    adapter.setDropDownViewResource(android.R.layout.simple_spinner_dropdown_item);
    placesSpinner.setAdapter(adapter);

    // Dugme za pretragu
    searchButton.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            String selectedPlaceType = placesSpinner.getSelectedItem().toString();
            searchNearbyPlaces(selectedPlaceType);
        }
    });
```

```
    // Inicijalizujte FusedLocationClient
    fusedLocationClient = LocationServices.getFusedLocationProviderClient(this);
}

@Override
public void onMapReady(GoogleMap googleMap) {
    mMap = googleMap;

    // Zatraži trenutnu lokaciju
    getCurrentLocation();
}

private void getCurrentLocation() {
    Task<android.location.Location> locationTask =
                                    fusedLocationClient.getLastLocation();
    locationTask.addOnSuccessListener(this, new
           OnSuccessListener<android.location.Location>() {
        @Override
        public void onSuccess(android.location.Location location) {
            if (location != null) {
                LatLng userLocation = new LatLng(location.getLatitude(),
                                        location.getLongitude());
                mMap.moveCamera(CameraUpdateFactory.newLatLngZoom(userLocation, 15));
            }
        }
    });
}

private void searchNearbyPlaces(String placeType) {
    // Pretraga prema izabranom tipu mesta (restoran, tržni centar, teretana)
    String placeTypeCode = getPlaceTypeCode(placeType);

    if (placeTypeCode != null) {
        // Koristite Places API za pretragu
        // (Pretraga sa odgovarajućim filterom za tip mesta)
    }
}
```

```
    private String getPlaceTypeCode(String placeType) {
        switch (placeType) {
            case "Restaurant":
                return "restaurant";
            case "Shopping Mall":
                return "shopping_mall";
            case "Gym":
                return "gym";
            default:
                return null;
        }
    }
}
```

Završite samostalno ovu vežbu i proširite je još nekim opcijama iz padajućeg menija. Treba da sadrži bar pet različitih tipova lokacija. Ubacite i notifikacije ili Toast poruke kada korisnik uspešno izlista neki konkretan tip objekta, ili kada nije ništa odabrao a kliknuo je na dugme da se prikaže. Možete implementirati i dugme koje vraća u padajućoj listi podrazumevanu selekciju i resetuje mapu, tj. sklanja sve što je obeleženo i samo se korisnik na trenutnoj lokaciji vidi.

Korisni resursi

1. Osnovno o lokacijama
   https://developer.android.com/develop/sensors-and-location/location
2. Malo detaljniji vodič sa primerom
   https://medium.com/@shaikvazid333/introduction-9c77c2719596
3. Koristan video tutorijal
   https://www.youtube.com/watch?v=M0kUd2dpxo4
4. Još jedan praktičan video tutorijal
   https://www.youtube.com/watch?v=H9WF3te9Gdw
5. Slično kao primer iz ove vežbe
   https://www.youtube.com/watch?app=desktop&v=gAIErOevfoA