**Tutorial para uso de mapas no Android – Google Maps**

Izabelle Lopes, Vinícius Carraro;

**O que é o Google Maps?**

Google Maps é um serviço de pesquisa e visualização de mapas e imagens de satélite da Terra gratuito para navegadores, iOS e Android fornecido e desenvolvido pela empresa estadunidense Google.

Para criar um aplicativo Android que utiliza o Google Maps, siga este tutorial passo a passo. Vamos configurar um projeto do zero, adicionar a biblioteca Google Maps, e criar uma simples exibição de mapa.

**Passo 1: Configurar um novo projeto Android**

1. **Inicie o Android Studio** e crie um novo projeto:
   * Selecione "New Project".
   * Escolha a opção "Google Maps Activity".
   * Configure o nome do projeto, nome do pacote, local de salvamento, linguagem (Java ou Kotlin) e defina o mínimo SDK (recomendado API 21 ou superior).

**Passo 2: Obter uma chave de API do Google Maps**

1. **Obter uma chave de API do Google Maps**:
   * Vá até o Google Cloud Platform Console.
   * Crie um novo projeto ou selecione um projeto existente.
   * Navegue até "APIs & Services" > "Credentials".
   * Clique em "Create credentials" e selecione "API key".
   * Copie a chave de API gerada.
2. **Habilitar APIs necessárias**:
   * Navegue até "APIs & Services" > "Library".
   * Habilite a "Maps SDK for Android".

**Passo 3: Configurar o projeto com a chave de API**

1. **Adicionar a chave de API no arquivo google\_maps\_api.xml**: O Android Studio gera este arquivo automaticamente quando você cria um projeto com a "Google Maps Activity". Abra o arquivo res/values/google\_maps\_api.xml e substitua YOUR\_API\_KEY pela sua chave de API:

xml

Copiar código

<resources>

<string name="google\_maps\_key" templateMergeStrategy="preserve" translatable="false">YOUR\_API\_KEY</string>

</resources>

1. **Configurar permissões no AndroidManifest.xml**: Certifique-se de que o arquivo AndroidManifest.xml contém as permissões necessárias e a chave da API:

xml

Copiar código

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

package="com.exemplo.googlemaps">

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

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

<application

android:allowBackup="true"

android:icon="@mipmap/ic\_launcher"

android:label="@string/app\_name"

android:roundIcon="@mipmap/ic\_launcher\_round"

android:supportsRtl="true"

android:theme="@style/AppTheme">

<meta-data

android:name="com.google.android.geo.API\_KEY"

android:value="@string/google\_maps\_key" />

<activity android:name=".MapsActivity">

<intent-filter>

<action android:name="android.intent.action.MAIN" />

<category android:name="android.intent.category.LAUNCHER" />

</intent-filter>

</activity>

</application>

</manifest>

**Passo 4: Exibir um mapa básico**

1. **Modificar a MapsActivity para exibir um mapa**: O Android Studio gera a MapsActivity automaticamente. Abra MapsActivity.java ou MapsActivity.kt e configure o mapa para exibir a localização do usuário:

java

Copiar código

// MapsActivity.java

package com.exemplo.googlemaps;

import androidx.fragment.app.FragmentActivity;

import android.os.Bundle;

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.maps.model.CameraPosition;

import android.Manifest;

import android.content.pm.PackageManager;

import androidx.core.app.ActivityCompat;

import androidx.core.content.ContextCompat;

public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {

private GoogleMap mMap;

private final int LOCATION\_REQUEST\_CODE = 101;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_maps);

// Obtain the SupportMapFragment and get notified when the map is ready to be used.

SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()

.findFragmentById(R.id.map);

mapFragment.getMapAsync(this);

}

@Override

public void onMapReady(GoogleMap googleMap) {

mMap = googleMap;

// Add a marker in Sydney and move the camera

LatLng sydney = new LatLng(-34, 151);

mMap.addMarker(new MarkerOptions().position(sydney).title("Marker in Sydney"));

mMap.moveCamera(CameraUpdateFactory.newLatLng(sydney));

// Check for location permission

if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_FINE\_LOCATION)

== PackageManager.PERMISSION\_GRANTED) {

mMap.setMyLocationEnabled(true);

} else {

ActivityCompat.requestPermissions(this, new String[]{Manifest.permission.ACCESS\_FINE\_LOCATION}, LOCATION\_REQUEST\_CODE);

}

}

@Override

public void onRequestPermissionsResult(int requestCode, String[] permissions, int[] grantResults) {

if (requestCode == LOCATION\_REQUEST\_CODE) {

if (permissions.length > 0 && grantResults[0] == PackageManager.PERMISSION\_GRANTED) {

if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS\_FINE\_LOCATION)

== PackageManager.PERMISSION\_GRANTED) {

mMap.setMyLocationEnabled(true);

}

}

}

}

}

1. **Layout XML para a MapsActivity**: Verifique se o arquivo activity\_maps.xml contém o fragmento do mapa:

xml

Copiar código

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

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

android:id="@+id/map"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

android:name="com.google.android.gms.maps.SupportMapFragment"/>

**Passo 5: Rodar o aplicativo**

1. **Executar no emulador ou dispositivo**: Conecte um dispositivo Android ou inicie um emulador e execute o aplicativo pelo Android Studio.

Pronto! Agora você tem um aplicativo básico do Android com um mapa interativo utilizando o Google Maps, que mostra a localização do usuário.