**Tutorial para uso de mapas no Android - MapBox**

Izabelle Lopes, Vinícius Carraro;

**O que é o MapBox?**

Mapbox é uma plataforma de localização ao vivo para mais de 1,6 milhão de desenvolvedores que fornece um mapa global, tráfego em tempo real, pesquisa de localização e navegação por meio de APIs e SDKs.

Para criar um aplicativo Android que utiliza o MapBox, siga este tutorial passo a passo. Vamos configurar um projeto do zero, adicionar a biblioteca MapBox, 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 "Empty 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).
2. **Configurar permissões no AndroidManifest.xml**: Abra o arquivo AndroidManifest.xml e adicione as seguintes permissões para permitir o acesso à Internet e localização:

xml

Copiar código

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

package="com.exemplo.mapbox">

<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">

<activity android:name=".MainActivity">

<intent-filter>

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

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

</intent-filter>

</activity>

</application>

</manifest>

**Passo 2: Adicionar o MapBox ao projeto**

1. **Adicionar dependências no build.gradle**: No arquivo build.gradle do módulo (geralmente app/build.gradle), adicione as dependências do MapBox:

groovy

Copiar código

dependencies {

implementation 'com.mapbox.maps:android:10.5.0' // versão pode variar

}

No mesmo arquivo, dentro do bloco android, adicione a seguinte configuração para habilitar a multidex (caso seu projeto exceda o limite de métodos do DEX):

groovy

Copiar código

android {

...

defaultConfig {

...

multiDexEnabled true

}

}

1. **Sincronize o projeto** para aplicar as mudanças.

**Passo 3: Inicializar o MapBox**

1. **Obter uma chave de API do MapBox**:
   * Vá até o site do [MapBox](https://www.mapbox.com/).
   * Crie uma conta ou faça login.
   * Navegue até a seção de Tokens de Acesso e crie um novo token para o seu aplicativo.
2. **Inicializar o MapBox na sua aplicação**: Abra o arquivo MainActivity.java ou MainActivity.kt e inicialize o MapBox no método onCreate:

java

Copiar código

// MainActivity.java

package com.exemplo.mapbox;

import android.os.Bundle;

import androidx.appcompat.app.AppCompatActivity;

import com.mapbox.maps.MapView;

import com.mapbox.maps.Style;

import com.mapbox.maps.plugin.locationcomponent.LocationComponentPlugin;

import com.mapbox.maps.plugin.locationcomponent.LocationComponentActivationOptions;

public class MainActivity extends AppCompatActivity {

private MapView mapView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

mapView = findViewById(R.id.mapView);

mapView.getMapboxMap().loadStyleUri(Style.MAPBOX\_STREETS, style -> {

// Style has been loaded, now you can add data or make other map adjustments

});

}

@Override

protected void onStart() {

super.onStart();

mapView.onStart();

}

@Override

protected void onStop() {

super.onStop();

mapView.onStop();

}

@Override

protected void onDestroy() {

super.onDestroy();

mapView.onDestroy();

}

@Override

protected void onPause() {

super.onPause();

mapView.onPause();

}

@Override

protected void onResume() {

super.onResume();

mapView.onResume();

}

}

**Passo 4: Layout XML**

1. **Adicionar MapView no layout XML**: Abra o arquivo activity\_main.xml e adicione o MapView:

xml

Copiar código

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

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

xmlns:tools="http://schemas.android.com/tools"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent"

tools:context=".MainActivity">

<com.mapbox.maps.MapView

android:id="@+id/mapView"

android:layout\_width="match\_parent"

android:layout\_height="match\_parent" />

</RelativeLayout>

**Passo 5: Configurar o mapa e exibir localização**

Para mostrar a localização do usuário, adicione o plugin de componente de localização:

1. **Configurar a localização do usuário no mapa**: Modifique o código da MainActivity para incluir a exibição da localização do usuário:

java

Copiar código

// MainActivity.java

package com.exemplo.mapbox;

import android.os.Bundle;

import androidx.annotation.NonNull;

import androidx.appcompat.app.AppCompatActivity;

import com.mapbox.maps.MapView;

import com.mapbox.maps.Style;

import com.mapbox.maps.plugin.Plugin;

import com.mapbox.maps.plugin.locationcomponent.LocationComponentPlugin;

import com.mapbox.maps.plugin.locationcomponent.LocationComponentActivationOptions;

import com.mapbox.maps.plugin.locationcomponent.OnLocationCameraTransitionListener;

import com.mapbox.maps.plugin.locationcomponent.OnLocationStaleListener;

import com.mapbox.maps.plugin.locationcomponent.modes.RenderMode;

import com.mapbox.maps.plugin.locationcomponent.modes.CameraMode;

public class MainActivity extends AppCompatActivity {

private MapView mapView;

@Override

protected void onCreate(Bundle savedInstanceState) {

super.onCreate(savedInstanceState);

setContentView(R.layout.activity\_main);

mapView = findViewById(R.id.mapView);

mapView.getMapboxMap().loadStyleUri(Style.MAPBOX\_STREETS, style -> {

initializeLocationComponent();

});

}

private void initializeLocationComponent() {

LocationComponentPlugin locationComponentPlugin = mapView.getPlugin(Plugin.Companion.getLocationComponent());

LocationComponentActivationOptions activationOptions = LocationComponentActivationOptions.builder(this, mapView.getMapboxMap().getStyle())

.useDefaultLocationEngine(true)

.build();

locationComponentPlugin.activateLocationComponent(activationOptions);

locationComponentPlugin.setLocationComponentEnabled(true);

locationComponentPlugin.setCameraMode(CameraMode.TRACKING);

locationComponentPlugin.setRenderMode(RenderMode.COMPASS);

}

@Override

protected void onStart() {

super.onStart();

mapView.onStart();

}

@Override

protected void onStop() {

super.onStop();

mapView.onStop();

}

@Override

protected void onDestroy() {

super.onDestroy();

mapView.onDestroy();

}

@Override

protected void onPause() {

super.onPause();

mapView.onPause();

}

@Override

protected void onResume() {

super.onResume();

mapView.onResume();

}

}

**Passo 6: 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 MapBox, que mostra a localização do usuário.