FUNDAÇÃO INSTITUTO DE EDUCAÇÃO DE BARUERI

Instituto Técnico de Barueri "Brasílio Flores de Azevedo"

Tecnologias Emergentes

Tutorial Geolocalização Mapa
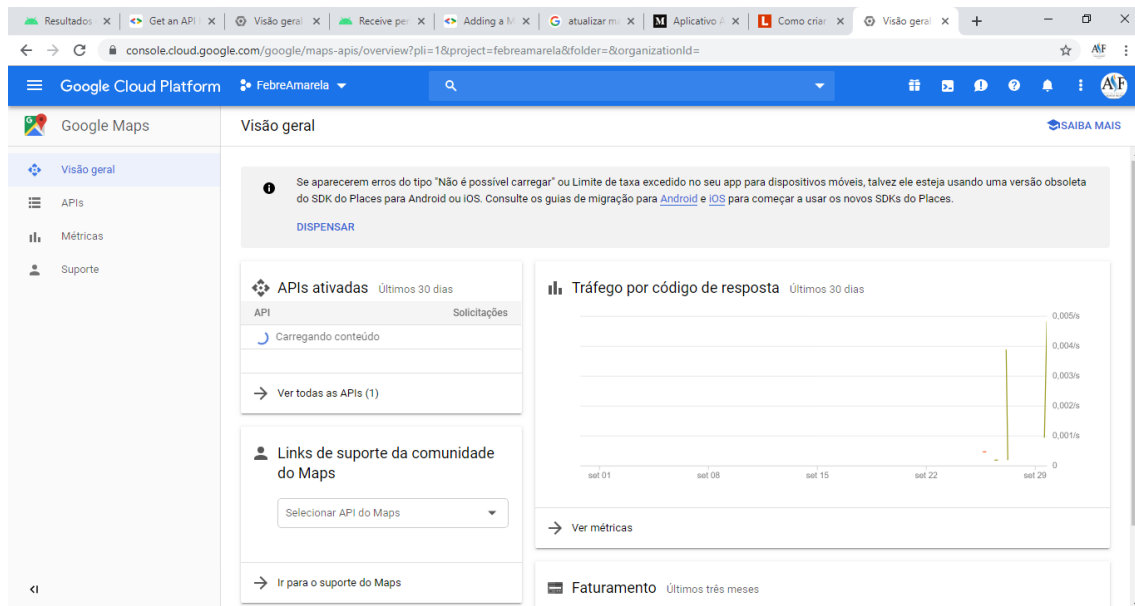
versão 1.0

Prof. Adriano Domingues

2019

AULA 18 – Geolocalização Carregando Mapa
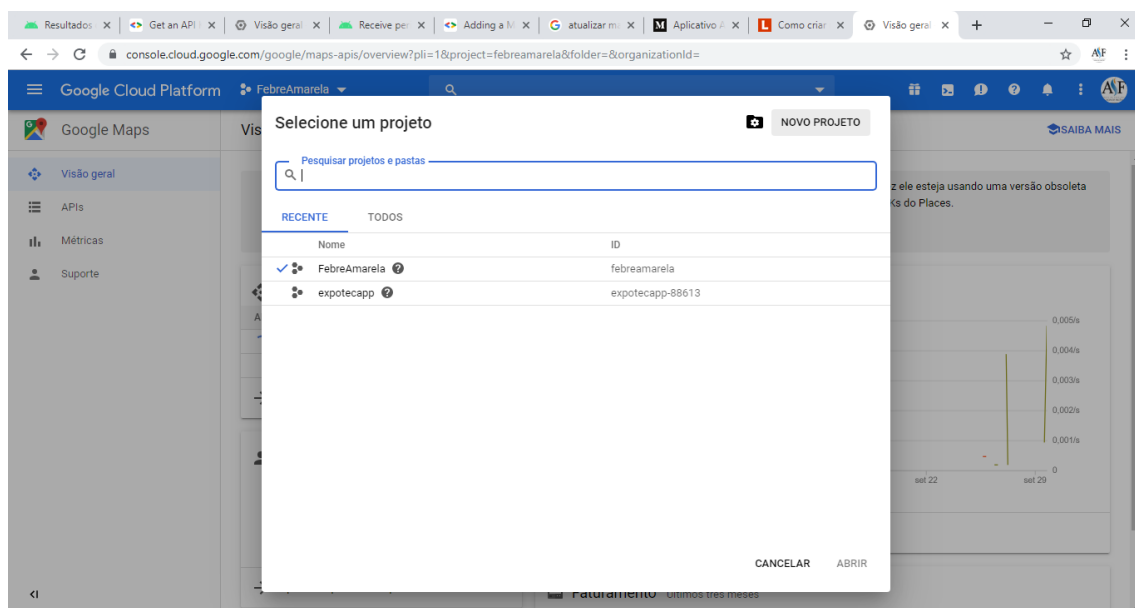
Vamos continuar no projeto Aula16Geo:

Para realizar este projeto você irá precisar da API key (chave de API), exigência da Google para que possamos utilizar o serviço de localização em um mapa.
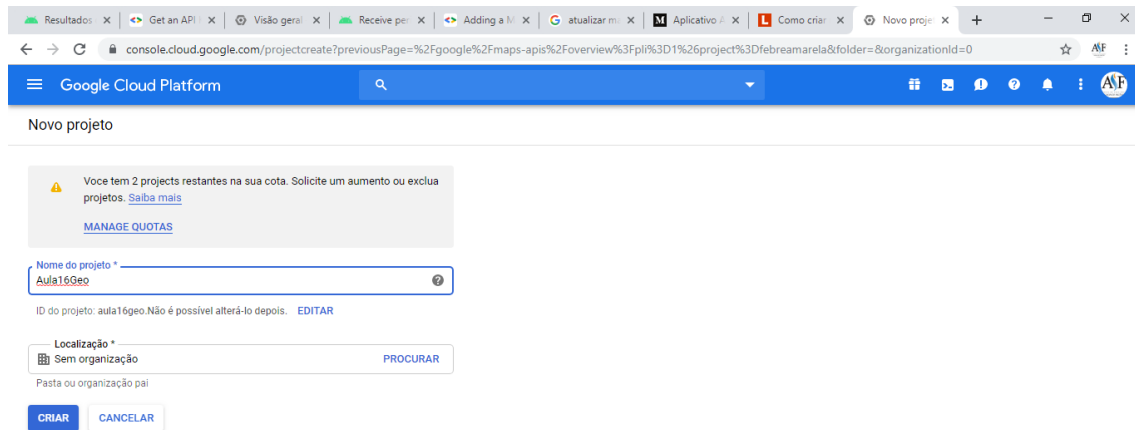
Com uma conta Google entre no site
https://console.cloud.google.com/google/maps-apis/overview
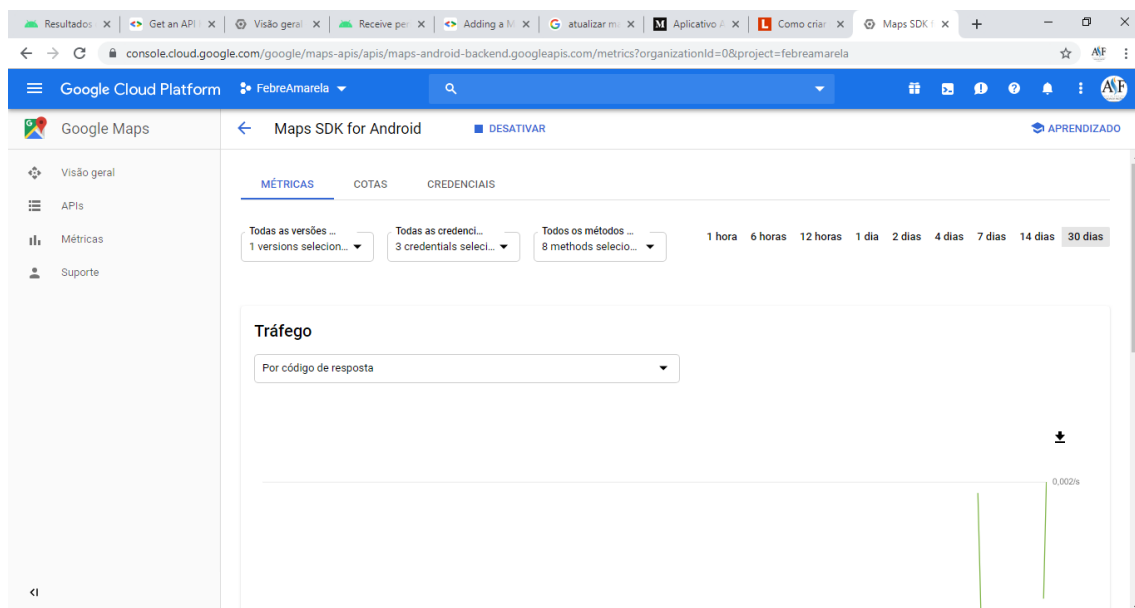

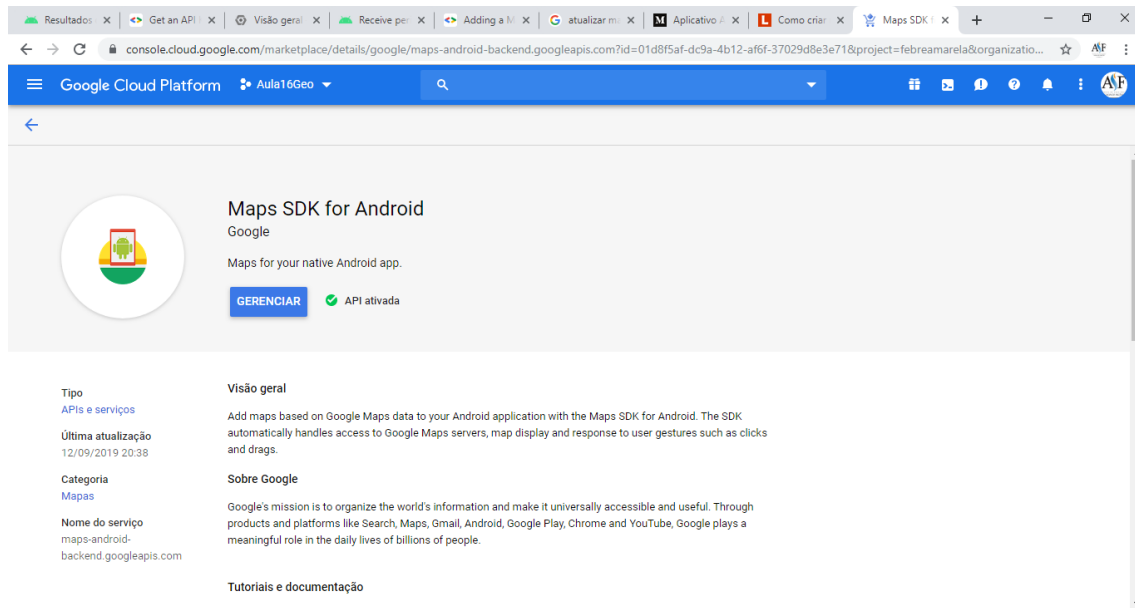Crie um novo projeto, no nosso caso o projeto se chama Aula16Geo:

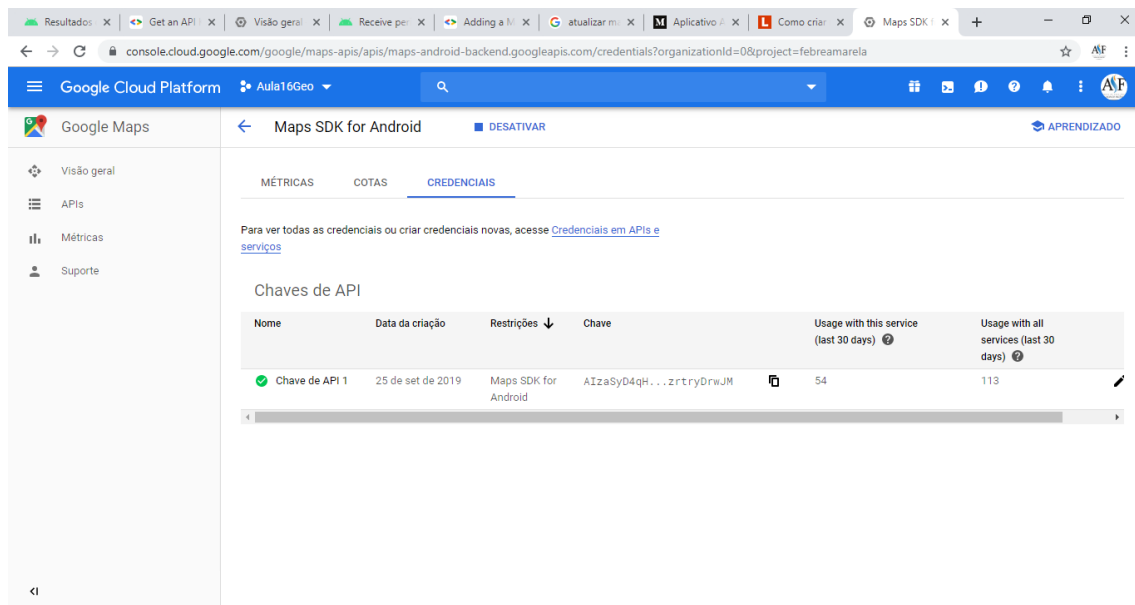Crie o projeto para possibilitar o vínculo de uma chave de API.

Em seguida clique em APIs e ative o *Maps SDK for Android*, realize a ativação do serviço.

Selecione o projeto criado Aula16Geo e clique no serviço Maps SDK for Android:

Clique no botão Gerenciar:

Em seguida clique no link *Credenciais em APIs e serviços*:
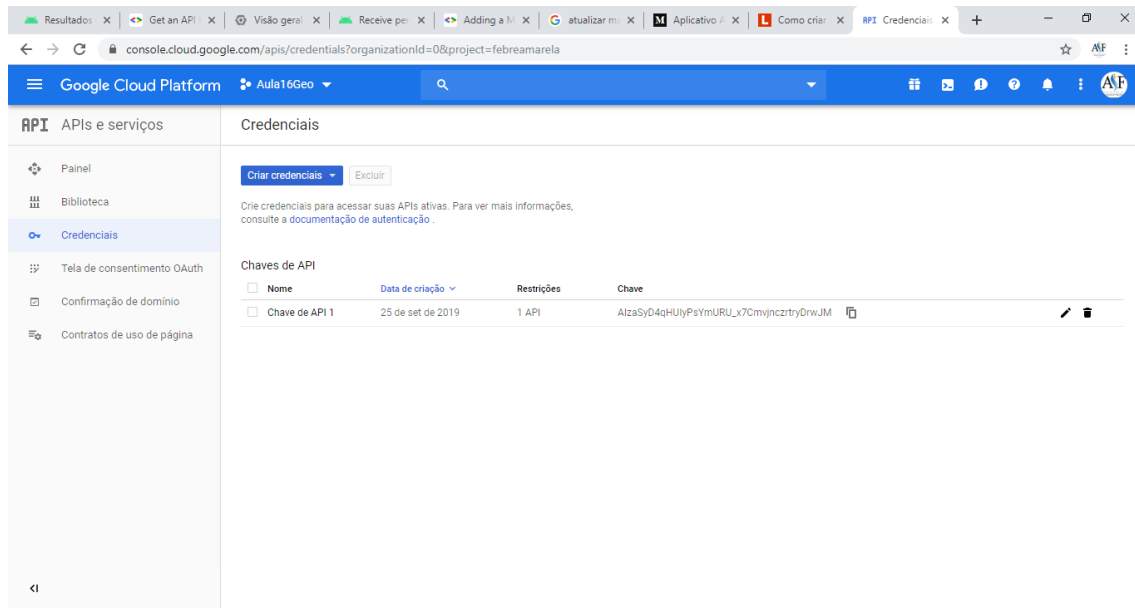
Cliquem em Criar credenciais > Chave de API

A chave será gerada:

Clique em Restringir Chave:

Configure as restrições:

Restrições de aplicativo > Apps para Android

Restringir o uso de apps para Android:

Precisamos preencher:

Nome do pacote
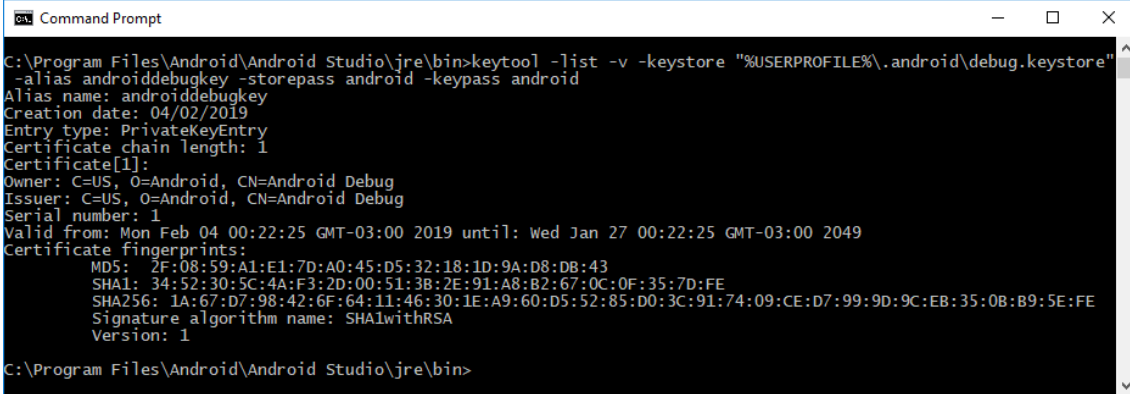
Impressão digital para certificação SHA-1

Este impressão digital é obtida executando o comando:

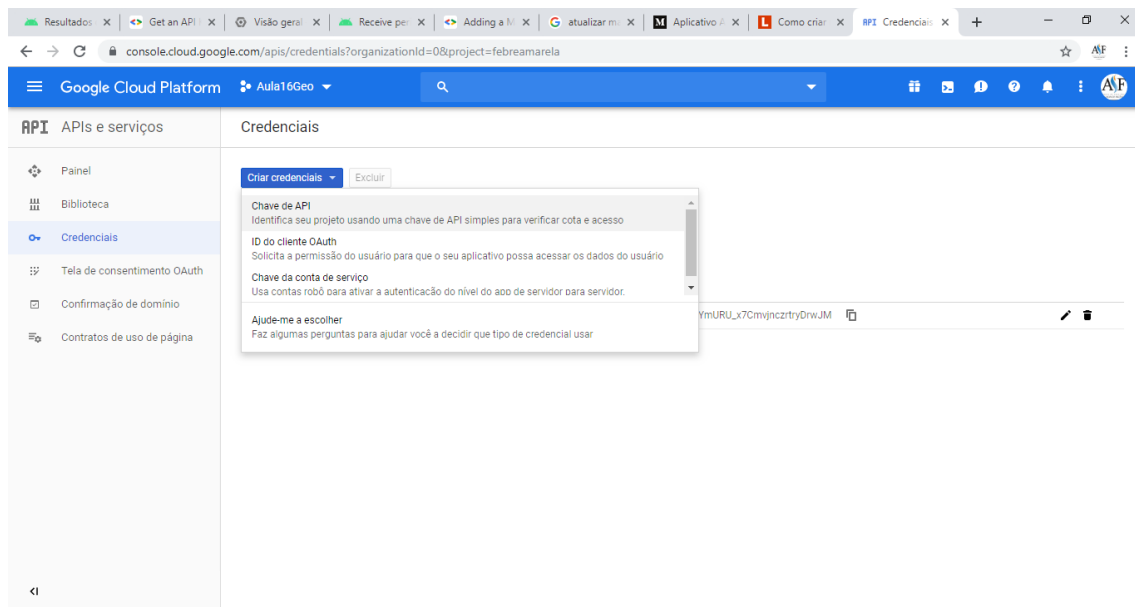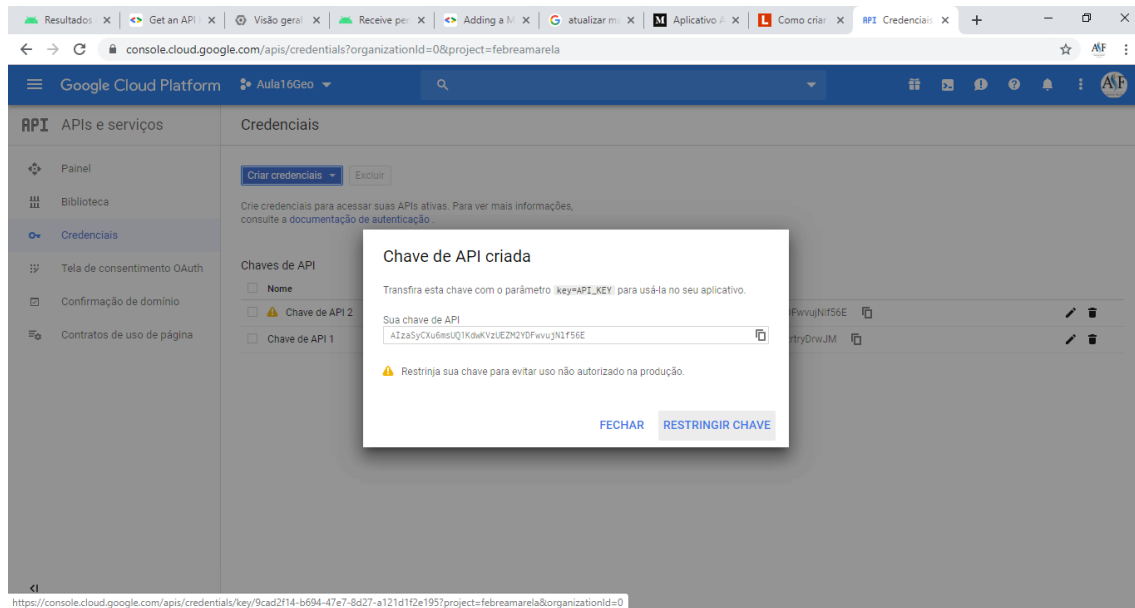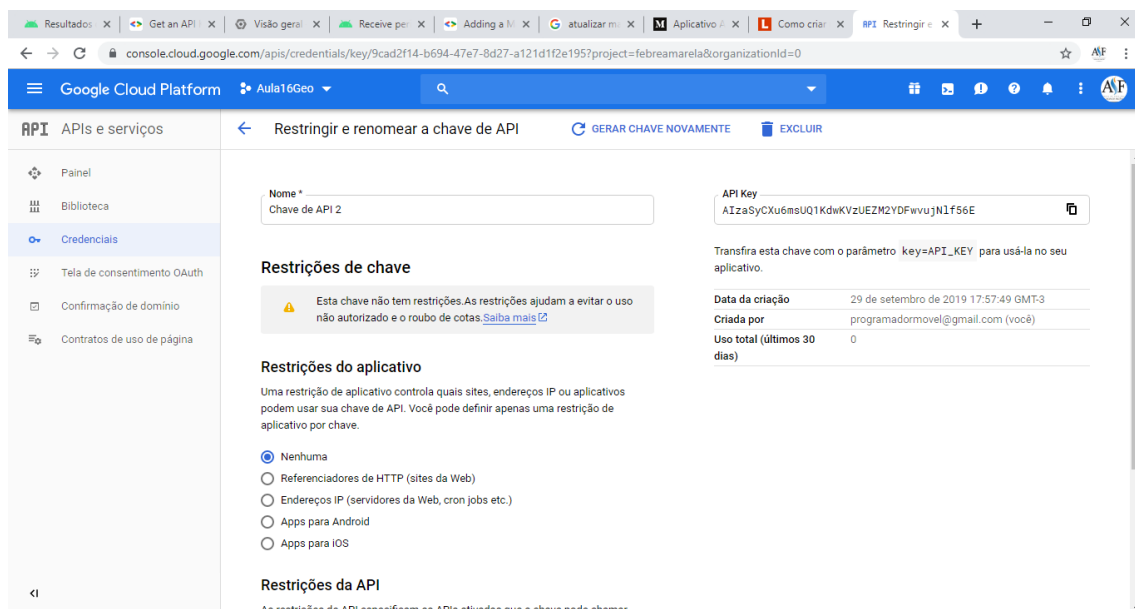keytool -list -v -keystore "%USERPROFILE%\.android\debug.keystore" -alias androiddebugkey -storepass android -keypass android

Abra o Prompt de Comando (cmd) e procure o executável keytool:

No meu caso o keytool esta na pasta C:\Program Files\Android\Android Studio\jre\bin>

Copie a chave SHA1 gerada e copie no campo exigido para restrição, e clique em Concluir.

Restrições de API > Restringir Chave > Maps SDK for Android

E clique em Salvar.
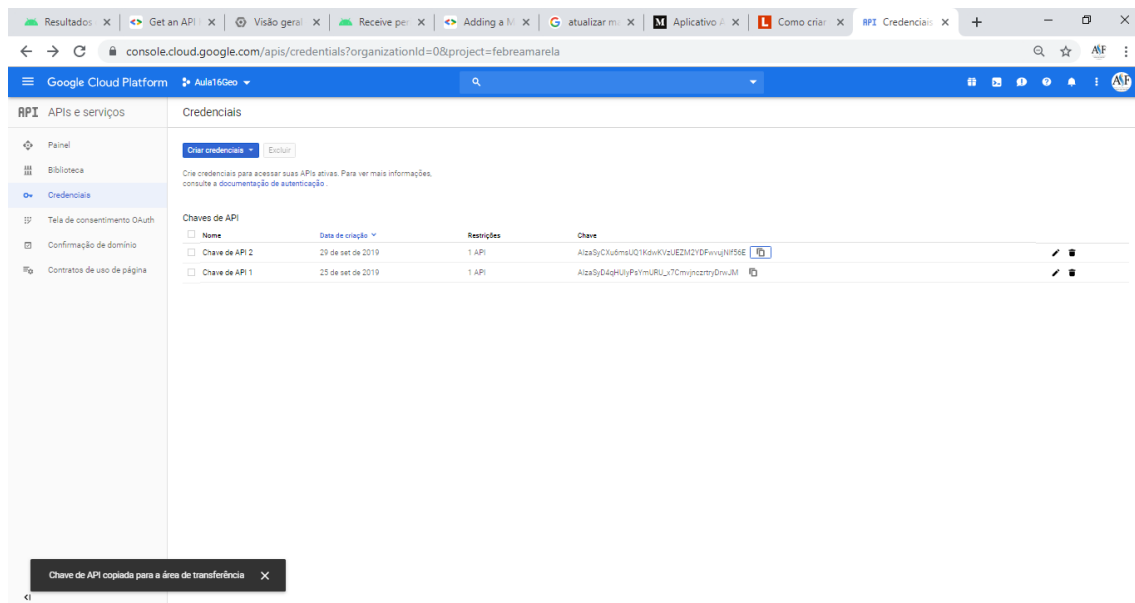
Copie a chave que será adicionada no Android Manifest.

Abra o arquivo AndroidManifest.xml em Android > app > manifests
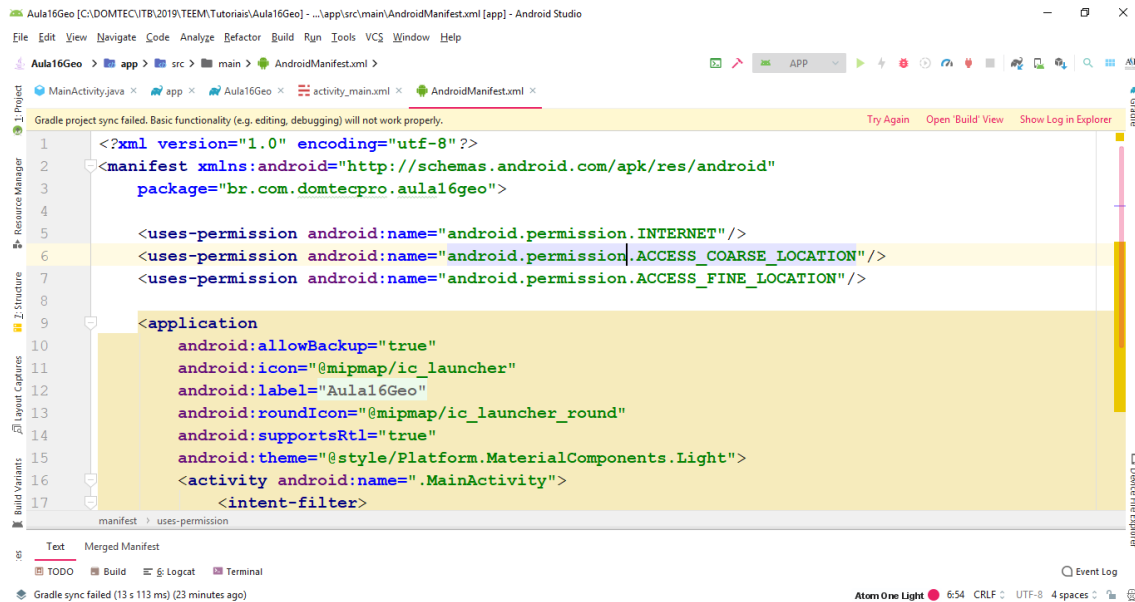
Abaixo da linha 15, antes da tag <activity> adicione:

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="YOUR_API_KEY"/>
```

No caso deste tutorial

```
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="AIzaSyCXu6msUQ1KdwKVzUEZM2YDFwvujNlf56E"/>
```
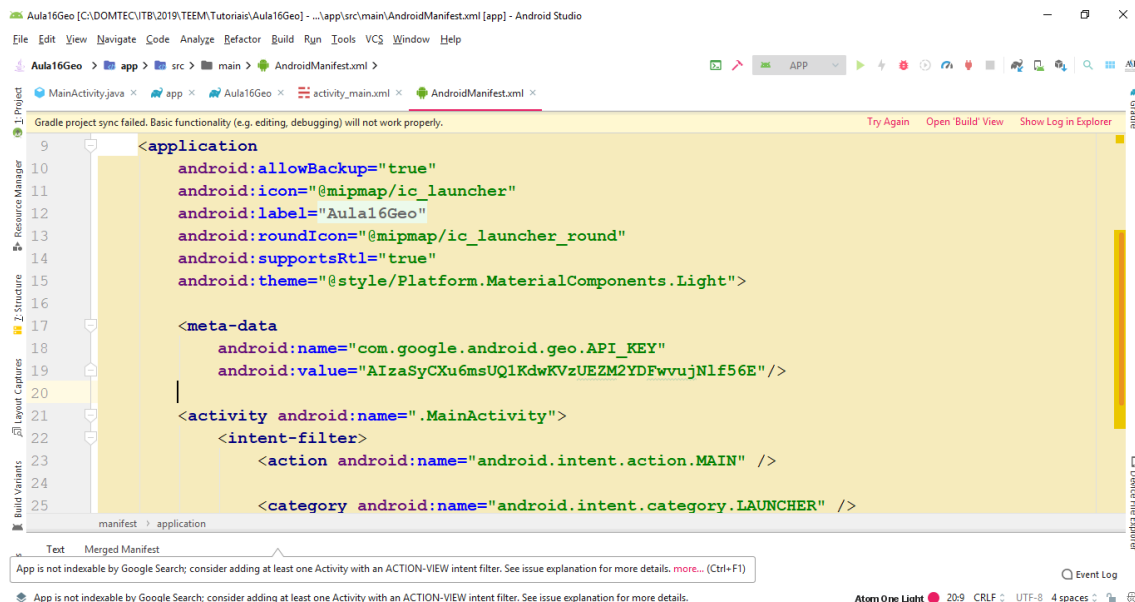

Na linha 21 em diante adicione esta meta-data que solicita o carregamento do Google Play Services no aparelho, de acordo com a versão compilada no projeto:

```xml
<meta-data
    android:name="com.google.android.gms.version"
    android:value="@integer/google_play_services_version" />
```
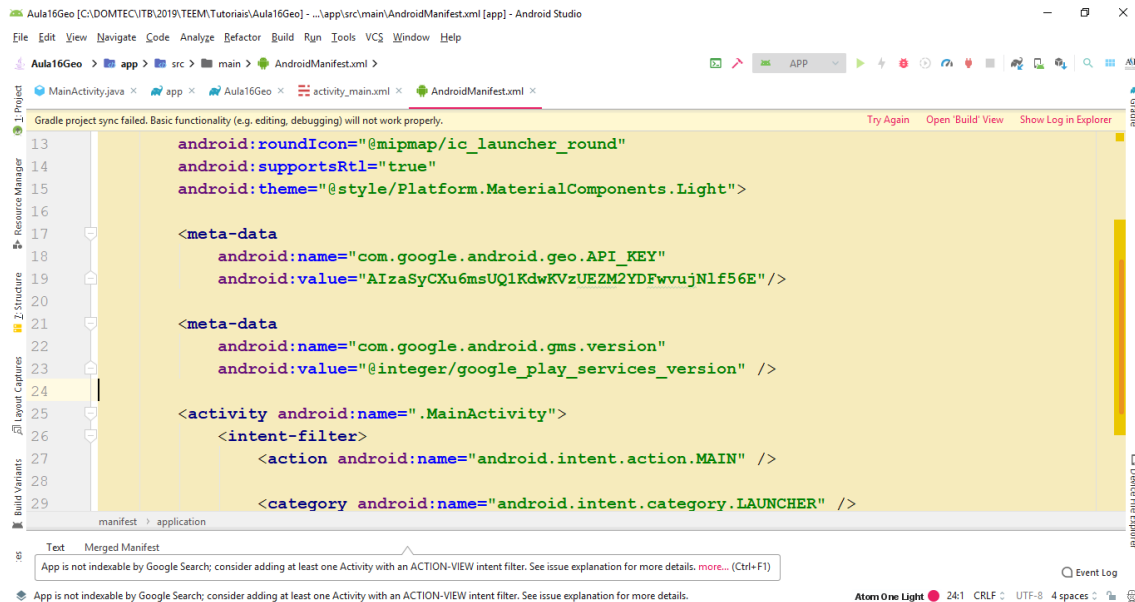

Adicione as permissões:

```xml
<uses-permission
    android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
<uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />
<uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />
```

Insira a declaração abaixo para tornar explícita a necessidade de utilização do OpenGL ES versão 2 para renderizar o mapa, se o aparelho não possuir este recurso o app não irá rodar adequadamente:

```xml
<uses-feature
    android:glEsVersion="0x00020000"
    android:required="true"/>
```

Ainda no Android Manifest adicione para definir a utilização da biblioteca Apache HTTP Legacy:

```xml
<uses-library
    android:name="org.apache.http.legacy"
    android:required="false" />
```
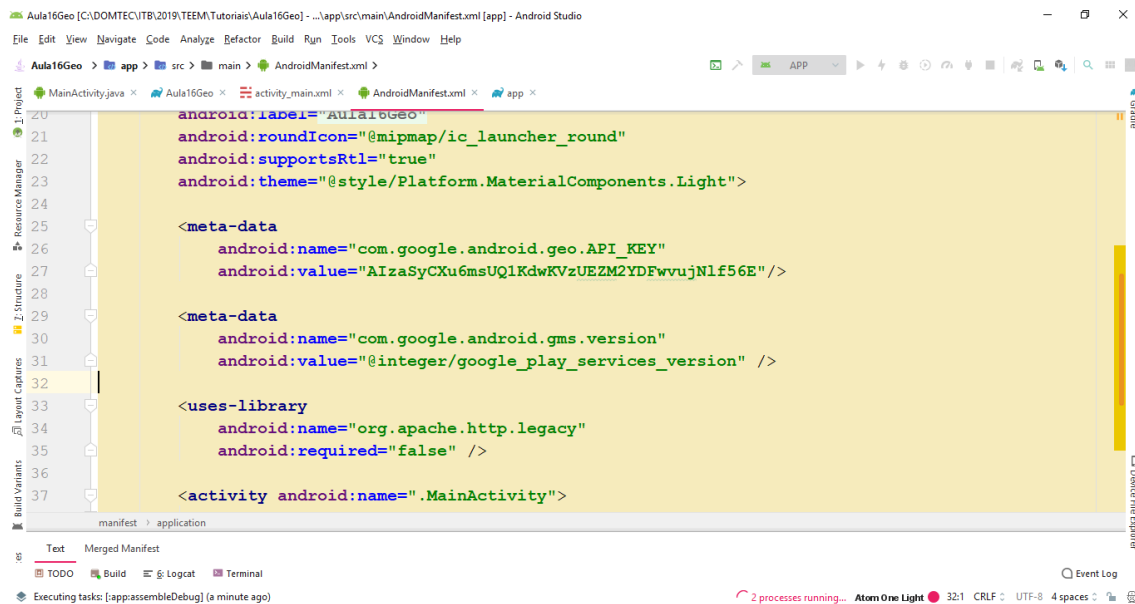
Copie os arquivos na pasta lib do projeto:

- maps-sdk-3.0.0-beta.aar
- places-maps-sdk-3.0.0-beta.aar

Altere o arquivo build.gradle (Project: Aula16Geo)

```
repositories {
    google()
    jcenter()
    flatDir {
        dirs 'libs'
    }
}
```
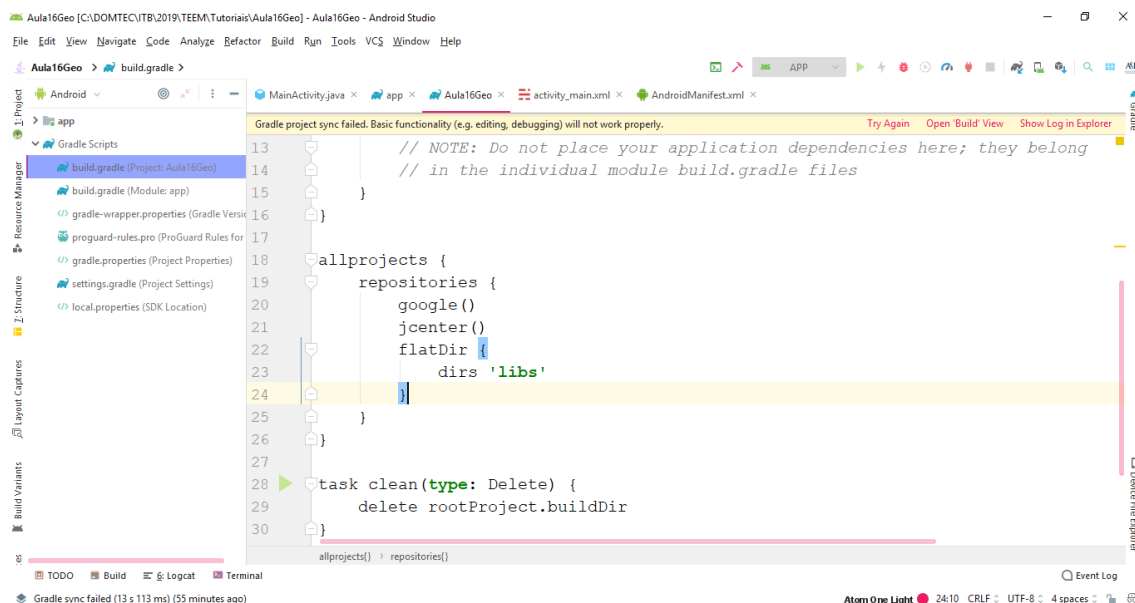
Adicione a dependência abaixo no build.gradle (Module: app):

```
implementation 'com.google.android.gms:play-services-maps:17.0.0'
implementation name:'maps-sdk-3.0.0-beta', ext:'aar'
implementation name:'places-maps-sdk-3.0.0-beta', ext:'aar'
implementation 'com.android.support:support-v4:+'
implementation 'com.android.support:appcompat-v7:+'
implementation 'com.google.android.gms:play-services-basement:15.0.1'
implementation 'com.google.android.gms:play-services-base:15.0.1'
implementation 'com.google.android.gms:play-services-gcm:15.0.1'
implementation 'com.google.android.gms:play-services-location:15.0.1'
implementation 'com.google.maps.android:android-maps-utils-sdk-v3-compat:0.1'
```

Ainda neste mesmo arquivo adicione o comando na linha 9:

```
multiDexEnabled true
```
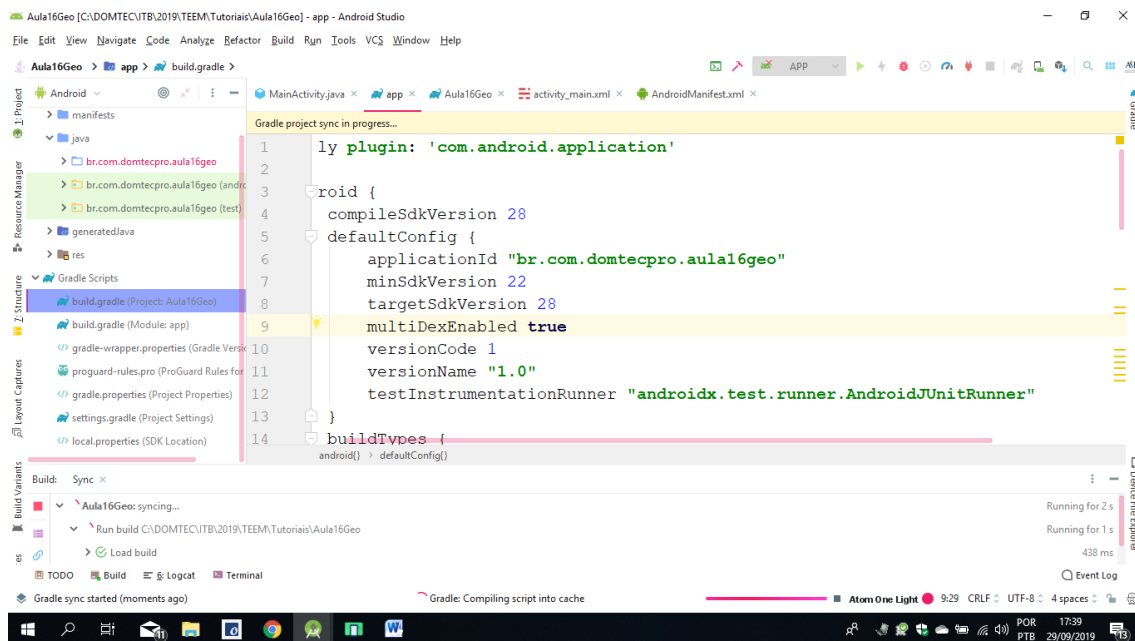
Nosso Android Manifest ficará assim:

```xml
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns:android="http://schemas.android.com/apk/res/android"
    package="br.com.domtecpro.aula16geo">

    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission
android:name="android.permission.ACCESS_COARSE_LOCATION"/>
    <uses-permission
android:name="android.permission.ACCESS_FINE_LOCATION"/>
    <uses-permission
        android:name="android.permission.WRITE_EXTERNAL_STORAGE" />
    <uses-permission
android:name="android.permission.READ_EXTERNAL_STORAGE" />
    <uses-permission
android:name="android.permission.ACCESS_NETWORK_STATE" />

    <uses-feature
        android:glEsVersion="0x00020000"
        android:required="true"/>

    <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/Platform.MaterialComponents.Light">

        <meta-data
            android:name="com.google.android.geo.API_KEY"
            android:value="AIzaSyCXu6msUQ1KdwKVzUEZM2YDFwvujNlf56E"/>

        <meta-data
            android:name="com.google.android.gms.version"
            android:value="@integer/google_play_services_version" />

        <uses-library
            android:name="org.apache.http.legacy"
            android:required="false" />

        <activity android:name=".MainActivity">
            <intent-filter>
                <action android:name="android.intent.action.MAIN" />

                <category android:name="android.intent.category.LAUNCHER"
/>
            </intent-filter>
        </activity>
    </application>
```

Sincronize o projeto e aguarde o download das bibliotecas no projeto.

Adicione o objeto do tipo MapView no layout do projeto (activity_main.xml):

```xml
<com.google.android.libraries.maps.MapView
    android:id="@+id/mapView"
    android:layout_width="0dp"
    android:layout_height="0dp"
    android:layout_marginStart="8dp"
    android:layout_marginTop="8dp"
    android:layout_marginBottom="8dp"
    android:apiKey="AIzaSyD4qHUIyPsYmURU_x7CmvjnczrtryDrwJM"
    android:clickable="true"
    android:enabled="true"
    android:focusable="auto"
    android:focusableInTouchMode="true"
    android:paddingLeft="10dp"
    android:paddingRight="10dp"
    app:layout_constraintBottom_toTopOf="@+id/chip"
    app:layout_constraintEnd_toEndOf="parent"
    app:layout_constraintStart_toStartOf="parent"
    app:layout_constraintTop_toTopOf="parent" />
```

Efetuando as alterações do layout temos:
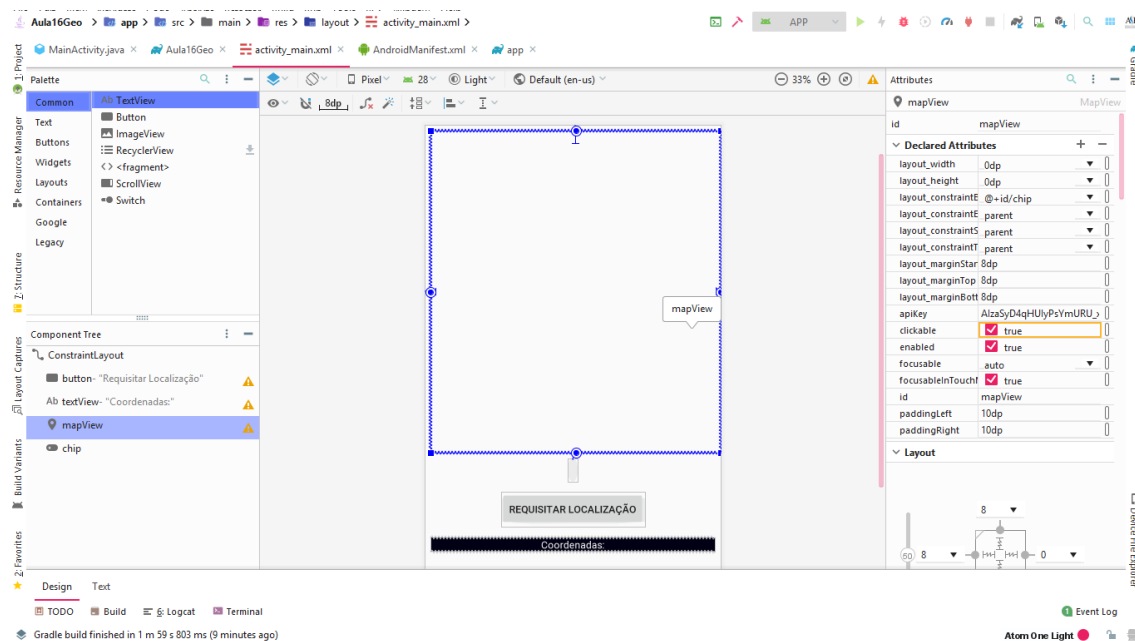
Na linha 45 faça as declarações dos objetos abaixo:

```java
MapView mapView;
Bundle mapViewBundle;
private GoogleMap gmap;
private static final String MAP_VIEW_BUNDLE_KEY =
"AIzaSyD4qHUIyPsYmURU_x7CmvjnczrtryDrwJM";
```

Na linha 52 insira o código para definir as coordenadas que serão apontadas no mapa:

```
// posição geográfica da cidade de Nova York
double latitude = 40.7143528, longitude = -74.0059731;
```

Vamos inserir a criação do mapa na linha 62:

```
mapView = findViewById(R.id.mapView);
```

E logo abaixo o pacote que armazena o credenciamento do mapa para acesso ao serviço do Google Maps:

```
mapViewBundle = null;
if (savedInstanceState != null) {
    mapViewBundle = savedInstanceState.getBundle(MAP_VIEW_BUNDLE_KEY);
}
```

A partir da linha 139, ainda dentro do método onCreate() insira a chamada do método de carregamento do mapa:

```
// Carrega o mapa com as coordenadas (latitude e longitude)
carregaMapa(mapViewBundle);
```

Na linha 144 vamos inserir o método de carregamento do mapa:

```
public void carregaMapa(Bundle mapViewBundle){
    mapView.setClickable(true);
    mapView.onCreate(mapViewBundle);
    mapView.getMapAsync(this);
}
```

Para que o mapa carregue as coordenadas da localização atual, devemos inserir a atualização da latitude e longitude no método onLocationChanged() do locationListener:

```
latitude = location.getLatitude();
longitude = location.getLongitude();
```
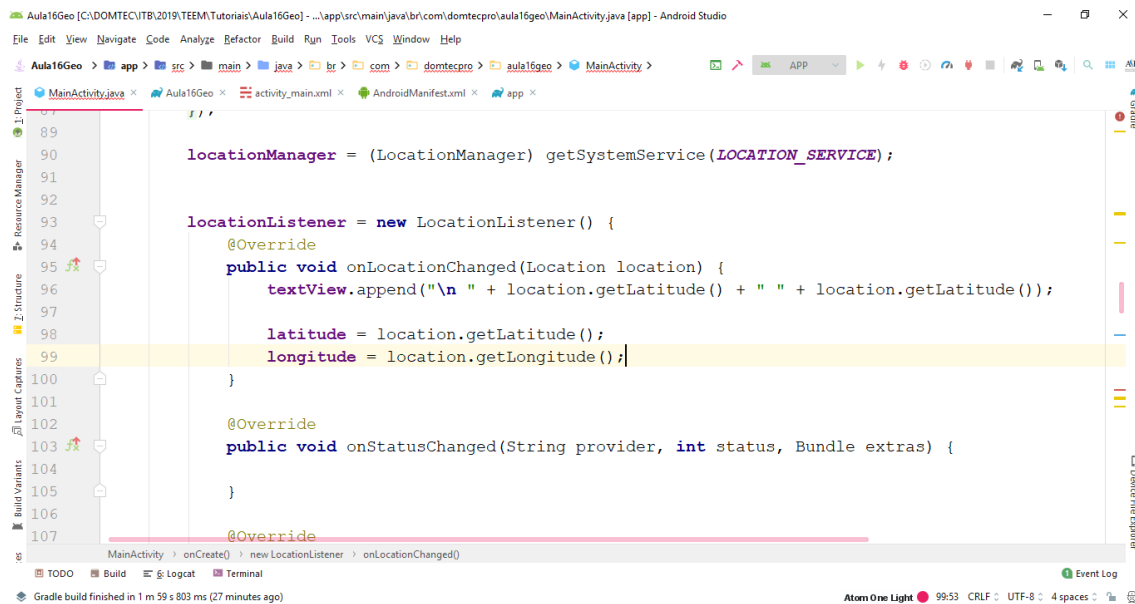
E na chamada do botão de requisição vamos adicionar o carregamento do mapa:

```java
private void configureButton() {
    button.setOnClickListener(new View.OnClickListener() {
        @Override
        public void onClick(View v) {
            try {
                locationManager.requestLocationUpdates("gps", 5000,
                        0, locationListener);
                chip.setCheckedIconVisible(true);
                carregaMapa(mapViewBundle);
            } catch (SecurityException e) {
                e.printStackTrace();
            }
        }
    });
}
```
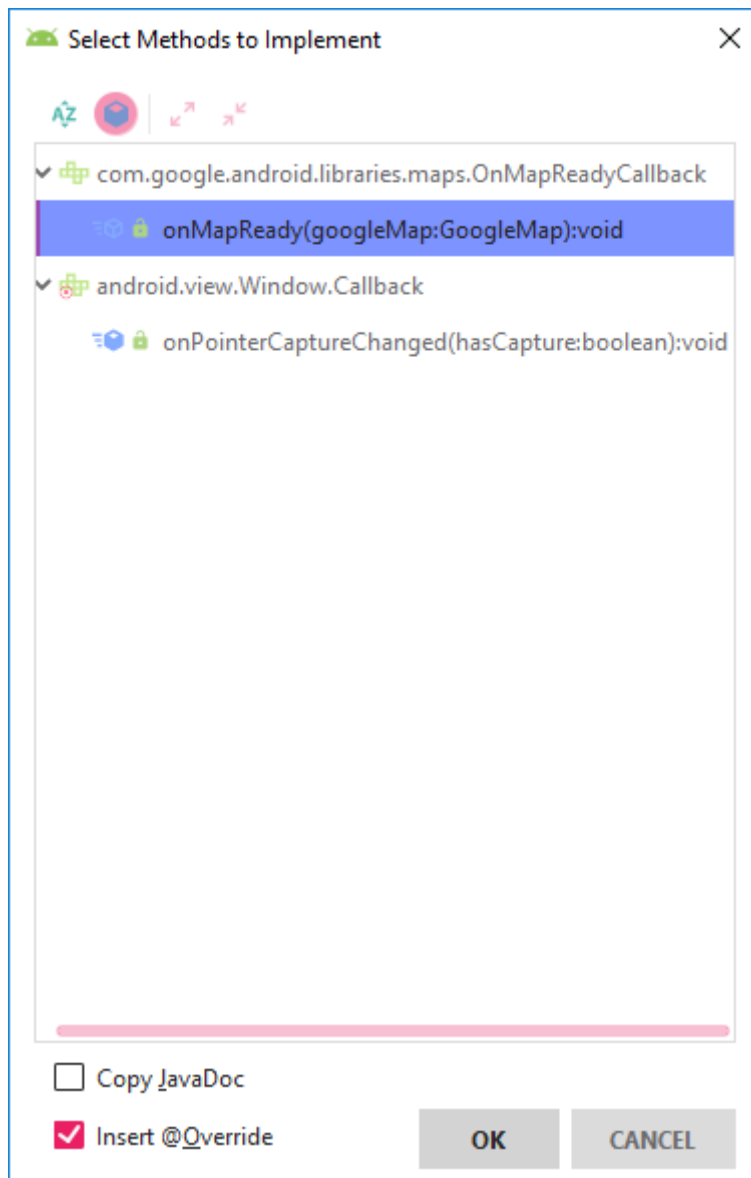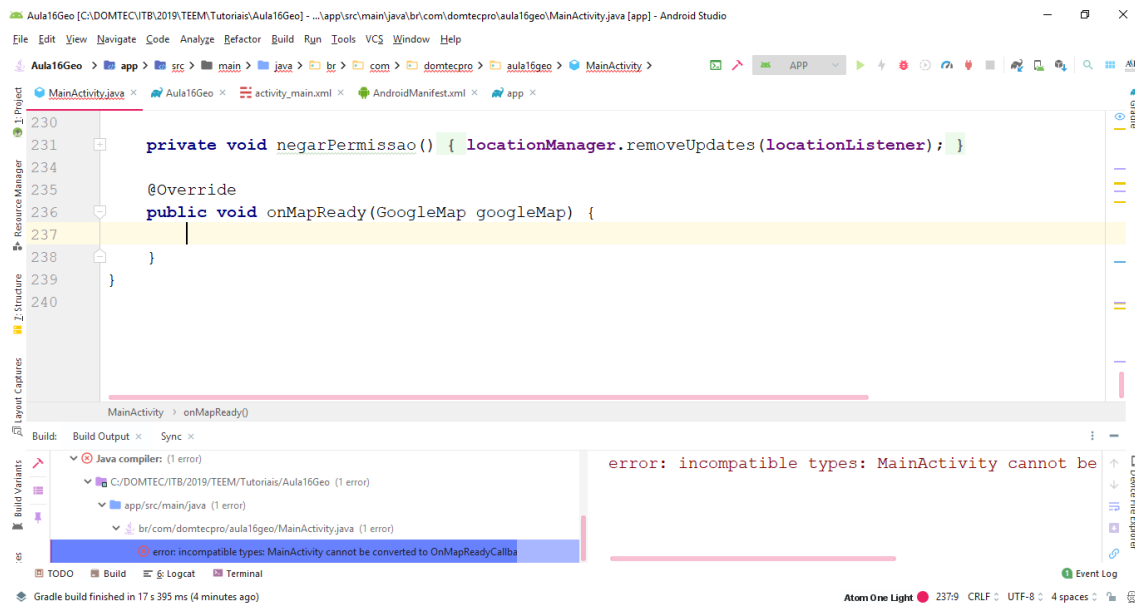
Por fim, precisamos adicionar a implementação do OnMapReadyCallBack na assinatura da classe MainActivity:

```java
public class MainActivity extends AppCompatActivity implements
OnMapReadyCallback {
```

E quando a linha ficar vermelha clique na lâmpada e selecione a opção **Implement Method:**

Clique em OK

E altere o método onMapReady() como abaixo:

```java
@Override
public void onMapReady(GoogleMap googleMap) {
    gmap = googleMap;
    gmap.setMinZoomPreference(12);
    LatLng ny = new LatLng(latitude, longitude);
    gmap.addMarker(new MarkerOptions().position(ny)
            .title("Localização Atual"));
    gmap.moveCamera(CameraUpdateFactory.
        newLatLngZoom(ny, 11));
    gmap.setTrafficEnabled(true);
}
```

Teste a aplicação e verifique se o mapa é carregado.

Adicione o ciclo de vida da atividade para controlar explicitamente o ciclo de vida do mapa:

```java
@Override
    public void onSaveInstanceState(Bundle outState) {
        super.onSaveInstanceState(outState);

        Bundle mapViewBundle = outState.getBundle(MAP_VIEW_BUNDLE_KEY);
        if (mapViewBundle == null) {
            mapViewBundle = new Bundle();
            outState.putBundle(MAP_VIEW_BUNDLE_KEY, mapViewBundle);
        }

        mapView.onSaveInstanceState(mapViewBundle);
    }
    @Override
    protected void onResume() {
        super.onResume();
        mapView.onResume();
    }
```

```java
    @Override
    protected void onStart() {
        super.onStart();
        mapView.onStart();
    }

    @Override
    protected void onStop() {
        super.onStop();
        mapView.onStop();
    }
    @Override
    protected void onPause() {
        mapView.onPause();
        super.onPause();
    }
    @Override
    protected void onDestroy() {
        mapView.onDestroy();
        super.onDestroy();
    }
    @Override
    public void onLowMemory() {
        super.onLowMemory();
        mapView.onLowMemory();
    }
```

Referência Bibliográfica

[1] http://developer.android.com. Acessado em 28/04/2019.

[2] https://www.youtube.com/watch?v=QNb_3QKSmMk. Acessado em 19/09/2019.

[3] https://imasters.com.br/android/requisitando-localizacao-usuario-da-forma-correta-no-android. Acessado em 20/09/2019.

[4] https://developers.google.com/maps/documentation/android-sdk/v3-client-migration. Acessado em 29/09/2019.

[5] https://developers.google.com/maps/documentation/android-sdk/map-with-marker. Acessado em 29/09/2019.

[6] https://www.luiztools.com.br/post/como-criar-um-aplicativo-android-com-gps/. Acessado em 29/09/2019.

[7] https://www.zoftino.com/android-mapview-tutorial. Acessado em 29/09/2019.

[8] https://developers.google.com/maps/documentation/android-sdk/current-place-tutorial. Acessado em 29/09/2019.