



Asignatura: Desarrollo de Aplicaciones Móviles Nativas.

Tema: Mapas 1. La clase MapActivity.

La Clase MapActivity.

El mapeo se ha convertido en uno de los servicios más populares de Google, que permite mapear, desde una ubicación cercana hasta direcciones muy lejanas, con vistas e imágenes de satélite.

La integración de Google Maps en aplicaciones de terceros requiere la aceptación de un extenso conjunto de términos legales que se deben revisar detenidamente para determinar si su uso previsto no cumplirá ninguna cláusula. Existen otras opciones de fuentes de datos de mapas, como OpenStreetMap.

A partir de Android 1.5, Google Maps no forma parte estrictamente del SDK de Android. En cambio, es parte del complemento de API de Google, una extensión del SDK estándar, como en el caso de MapView y MapActivity. El sistema de complementos de Android proporciona enlaces para otros subsistemas, que pueden ser parte de algunos dispositivos, pero no de otros.

Para probar su integración de Google Maps, se necesita un AVD que utiliza un objetivo apropiado. Además, se deben generar claves propias de API para usar con la aplicación. Se pueden obtener instrucciones completas para generar claves API, para desarrollo y uso de producción. Para registrar la huella digital MD5 del certificado que se utilizará para firmar su solicitud, el servicio de registro de Maps proporciona una clave API de Maps que está asociada con el certificado de firmante de su aplicación. Enseguida, agregar una referencia a la clave de API de Maps en cada MapView, ya sea declarada en XML o instanciada directamente desde el código. Se puede utilizar la misma clave de API de Maps para cualquier MapView en cualquier aplicación de Android, siempre que la aplicación esté firmada con el certificado cuya huella digital se registró en el servicio.

NOTA: Debido a que Google Maps no forma parte del proyecto de código abierto de Android, algunos dispositivos carecen de Google Maps debido a problemas de licencia, pero ello no afecta su desarrollo.

DESARROLLO

PARTE I.

1. Crear un nuevo proyecto en Android Studio.
2. En la ventana **Create New Project**, seleccionar **Google Maps Activity**. Clic en **Next**.

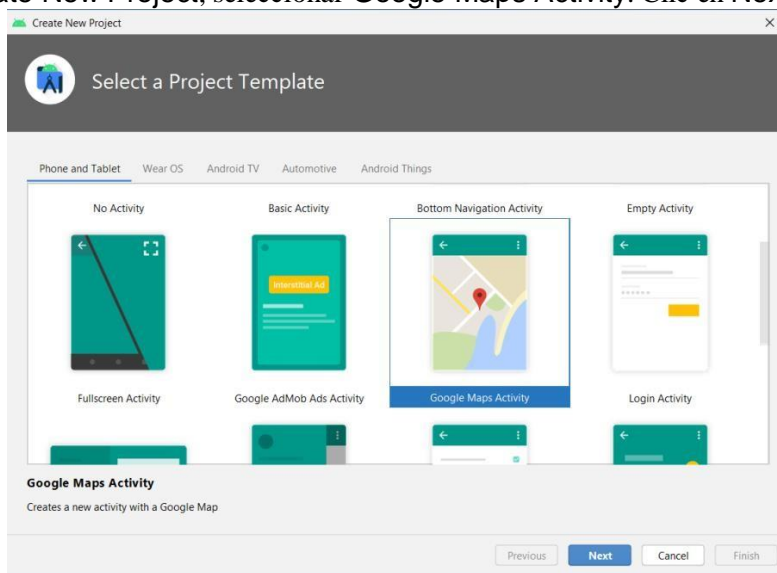


Figura 1. Seleccionar Google Maps Activity.



3. En la ventana Configure Your Project, se pueden cambiar algunas características, enseguida clic en Finish.

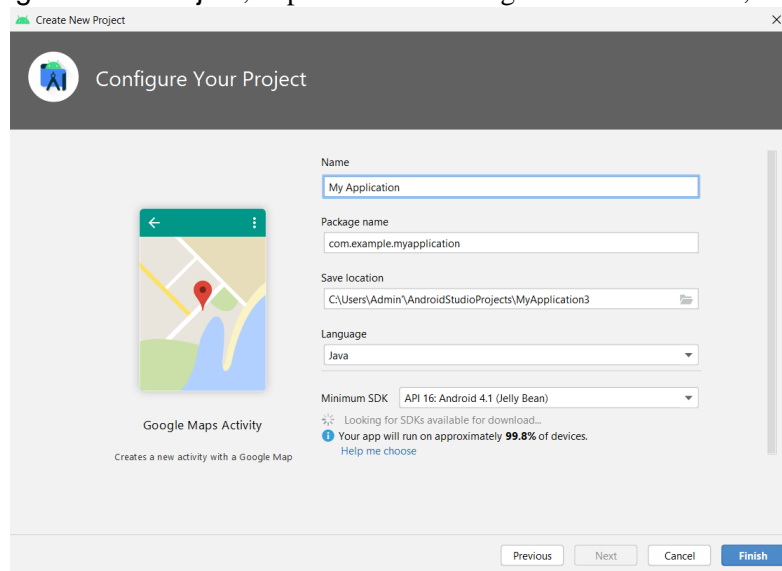


Figura 2. Seleccionar las características necesarias y clic en Finish.

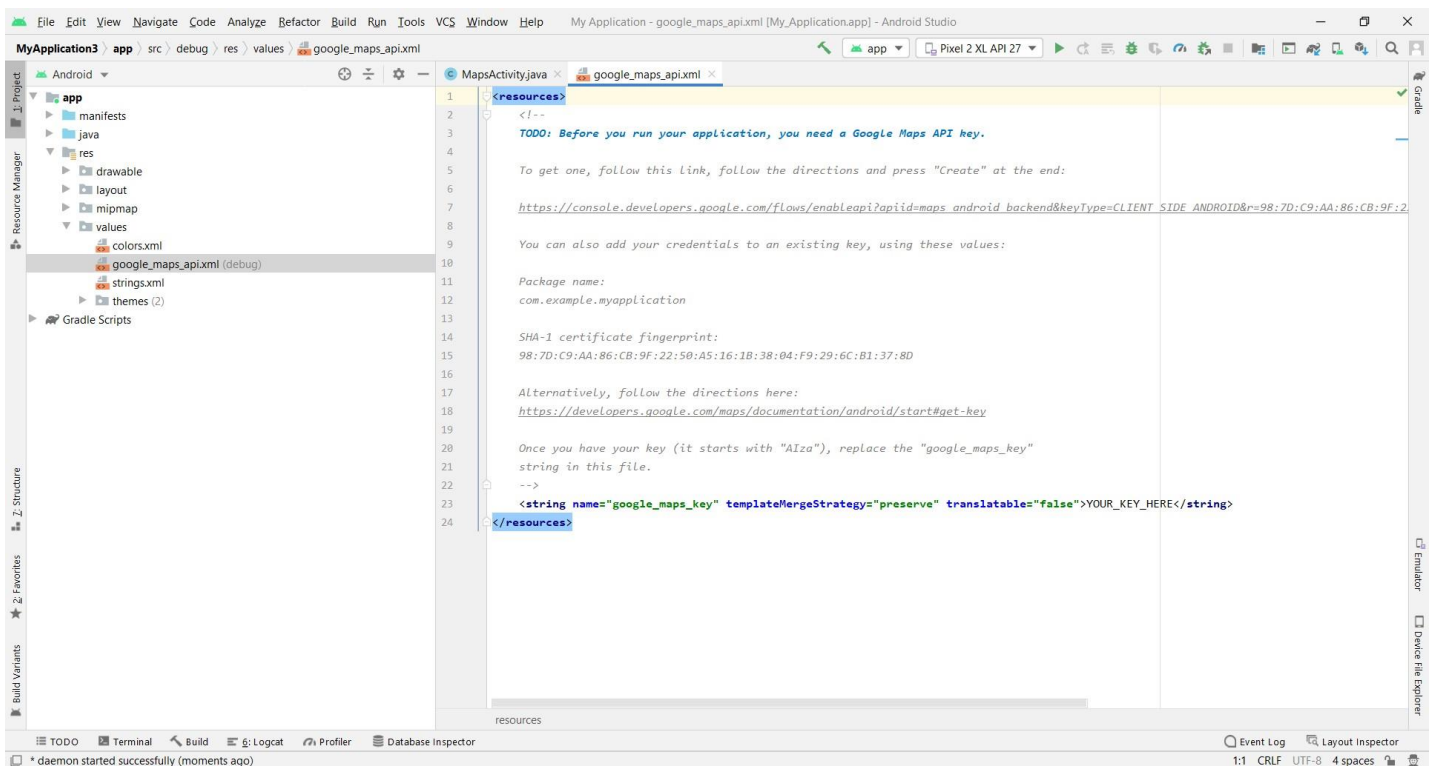


Figura 3. La ventana inicial muestra el contenido del archivo google_maps_api.xml.

4. Enseguida se muestran los contenidos de los archivos básicos del proyecto de mapas.

El archivo google_maps_api.xml:

```
<resources>
```

```
<!--
```

```
TODO: Before you run your application, you need a Google Maps API key.
```



To get one, follow this link, follow the directions and press "Create" at the end:

https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&keyType=CLIENT_SIDE_ANDROID&r=98:7D:C9:AA:86:CB:9F:22:50:A5:16:1B:38:04:F9:29:6C:B1:37:8D%3Bcom.example.myapplication

You can also add your credentials to an existing key, using these values:

Package name:

com.example.myapplication

SHA-1 certificate fingerprint:

98:7D:C9:AA:86:CB:9F:22:50:A5:16:1B:38:04:F9:29:6C:B1:37:8D

Alternatively, follow the directions here:

<https://developers.google.com/maps/documentation/android/start#get-key>

Once you have your key (it starts with "AIza"), replace the "google_maps_key" string in this file.

-->

```
<string name="google_maps_key" templateMergeStrategy="preserve"
translatable="false">YOUR_KEY_HERE</string>
</resources>
```

El archivo **MapsActivity.java**:

```
package com.example.myapplication;

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;

public class MapsActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;

    @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);
    }

    /**
     * Manipulates the map once available.
     */
}
```



```

    * This callback is triggered when the map is ready to be used.
    * This is where we can add markers or lines, add listeners or move the camera. In this case,
    * we just add a marker near Sydney, Australia.
    * If Google Play services is not installed on the device, the user will be prompted to
install
    * it inside the SupportMapFragment. This method will only be triggered once the user has
    * installed Google Play services and returned to the app.
    */
@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));
}
}

```

El archivo activity_maps.xml:

```

<?xml version="1.0" encoding="utf-8"?>
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
    xmlns:map="http://schemas.android.com/apk/res-auto"
    xmlns:tools="http://schemas.android.com/tools"
    android:id="@+id/map"
    android:name="com.google.android.gms.maps.SupportMapFragment"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
    tools:context=".MapsActivity" />

```

El archivo AndroidManifest.xml:

```

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

    <!--
        The ACCESS_COARSE/FINE_LOCATION permissions are not required to use
        Google Maps Android API v2, but you must specify either coarse or fine
        location permissions for the "MyLocation" functionality.
    -->

    <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/Theme.MyApplication">

        <!--
            The API key for Google Maps-based APIs is defined as a string resource.
            (See the file "res/values/google_maps_api.xml").

```



Note that the API key is linked to the encryption key used to sign the APK.
You need a different API key for each encryption key, including the release key that is used to sign the APK for publishing.
You can define the keys for the debug and release targets in `src/debug/` and `src/release/`.

```
-->

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

<activity
    android:name=".MapsActivity"
    android:label="@string/title_activity_maps">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />

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

</manifest>
```

5. Al ejecutar la aplicación se muestra la aplicación con un mapa predeterminado por las coordenadas de Australia. Sin embargo, si por alguna circunstancia se llega a mostrar una plantilla vacía se requiere realizar algunos cambios necesarios en la instalación de bibliotecas de Android Studio.

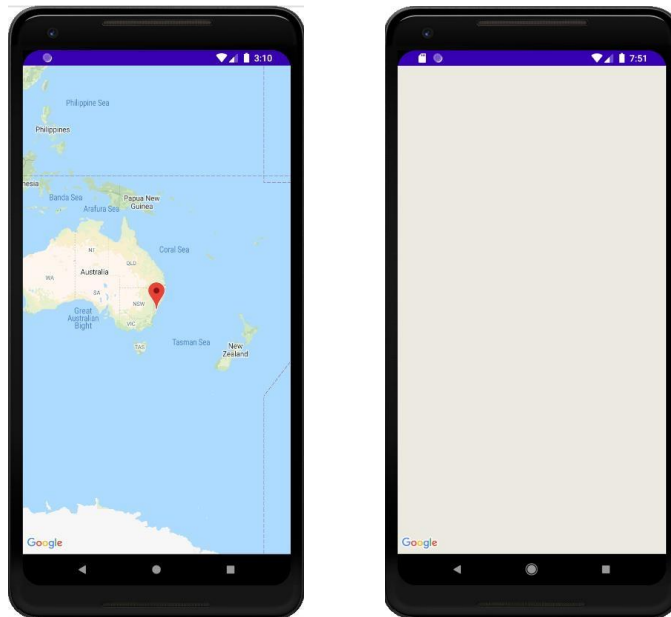


Figura 4. Se muestra el mapa de Australia y una aplicación inicial vacía.

6. Abrir la ventana **Settings for New Projects** y habilitar la opción de **Google Play services** para la descarga.

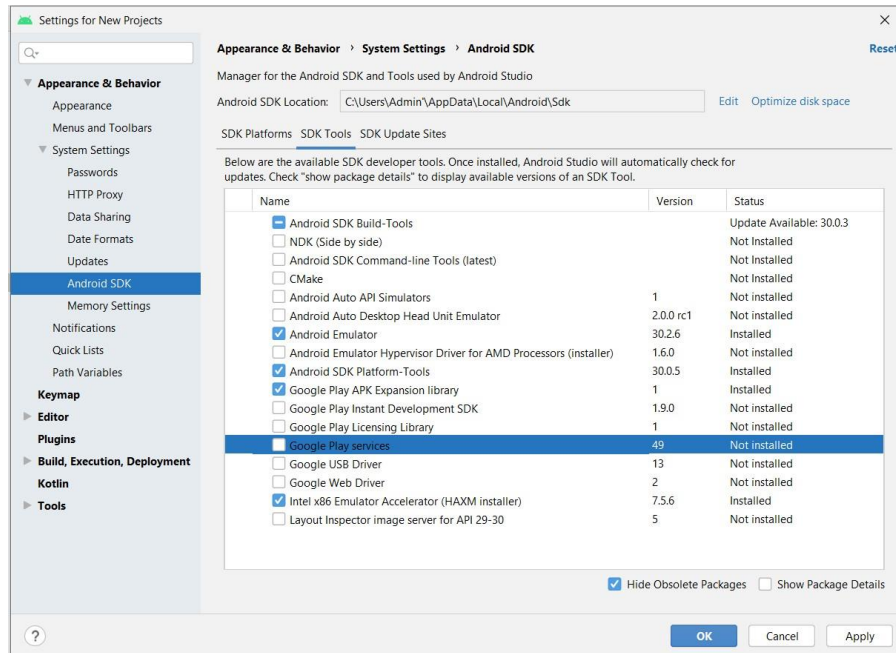


Figura 5. La ventana Settings for New Projects y la selección de Google Play services.

7. Confirmar la instalación de los componentes.

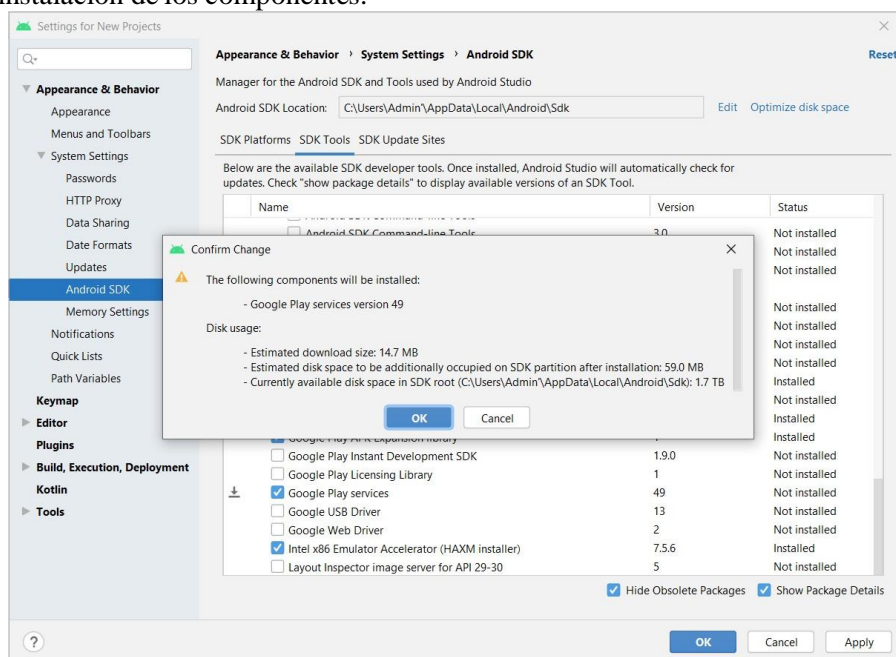


Figura 6. Digitar OK para confirmar la descarga.

8. Esperar la terminación y la instalación de la descarga. Al terminar digitar en Finish.

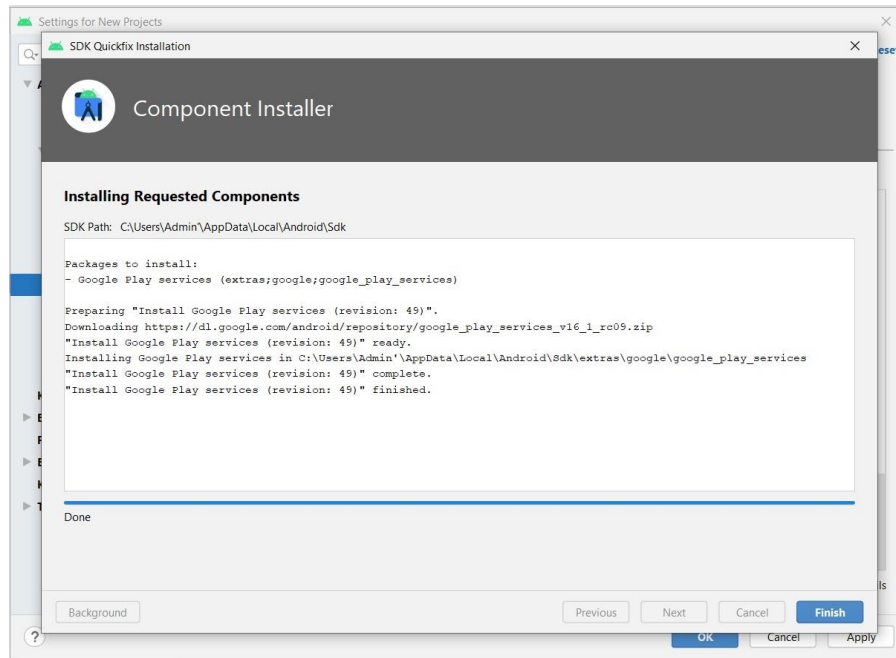


Figura 7. Digitar Finish para terminar la instalación de Google Play services.

PARTE II.

9. Ingresar al sitio indicado en el archivo **google_maps_api.xml** y crear la clave para el uso de mapas.

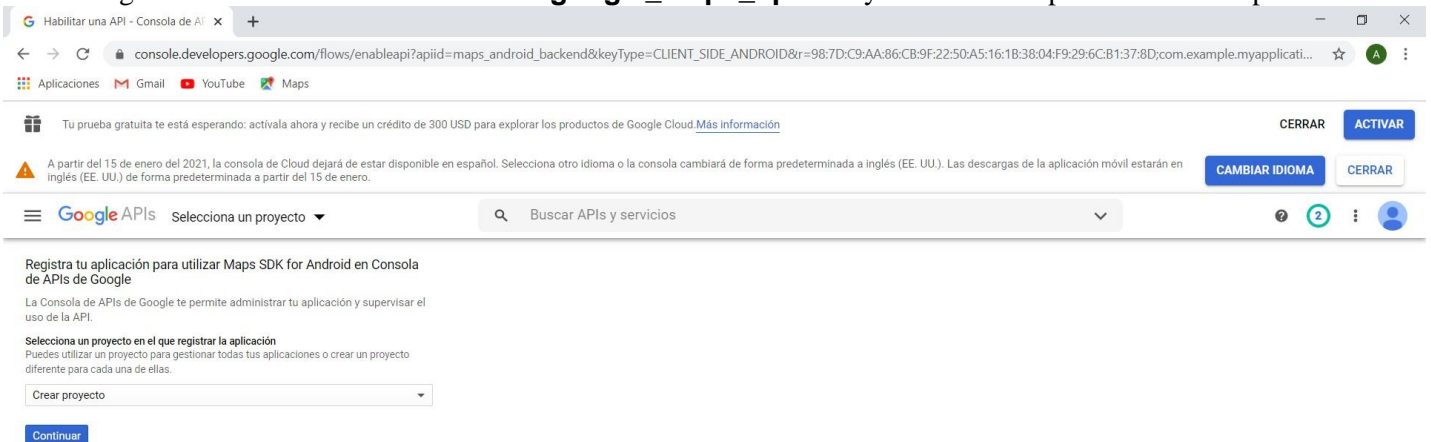


Figura 8. Ingreso al sitio sugerido en el archivo XML para la creación de claves:

https://console.developers.google.com/flows/enableapi?apiid=maps_android_backend&keyType=CLIENT_SIDE_ANDROID&r=98:7D:C9:AA:86:CB:9F:22:50:A5:16:1B:38:04:F9:29:6CB1:37:8D%3Bcom.example.myapplication

10. Seguir las indicaciones del sitio. Seleccionar un proyecto si es necesario, en este caso My Project.

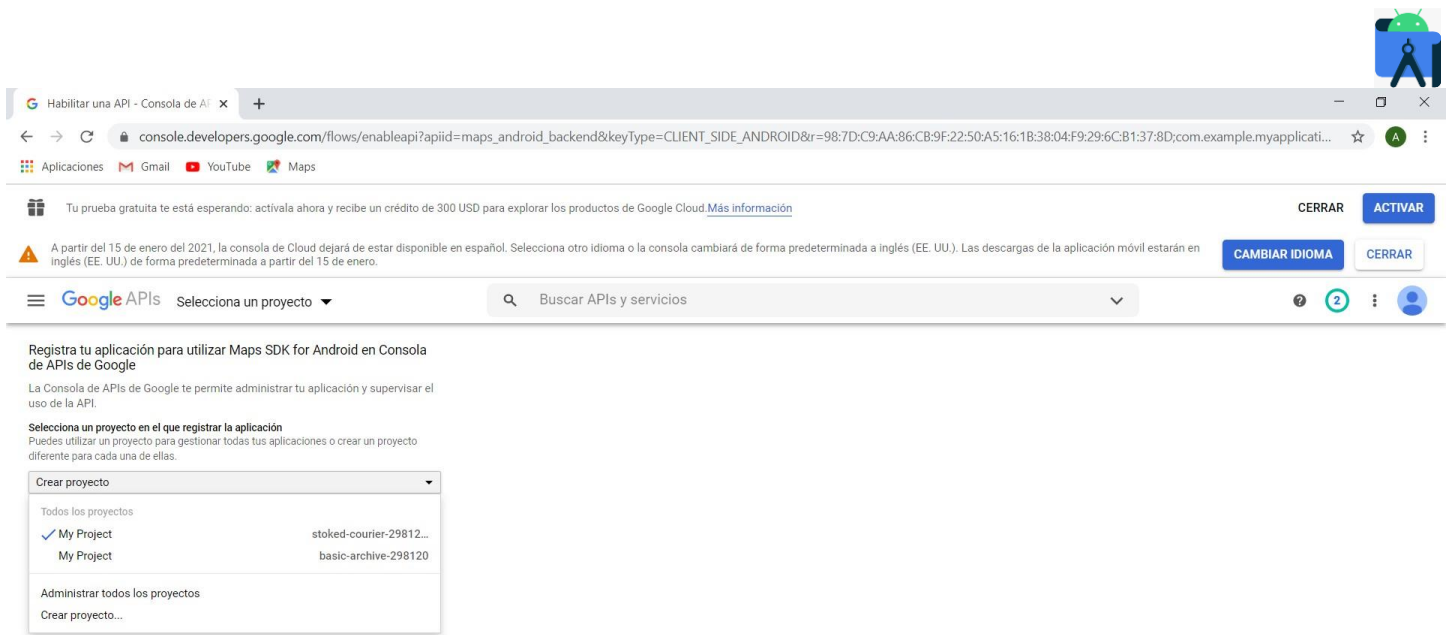


Figura 9. Seleccionar un proyecto para crear una clave.

11. Digitar el botón Crear clave de API para generar la clave para el proyecto de mapas.

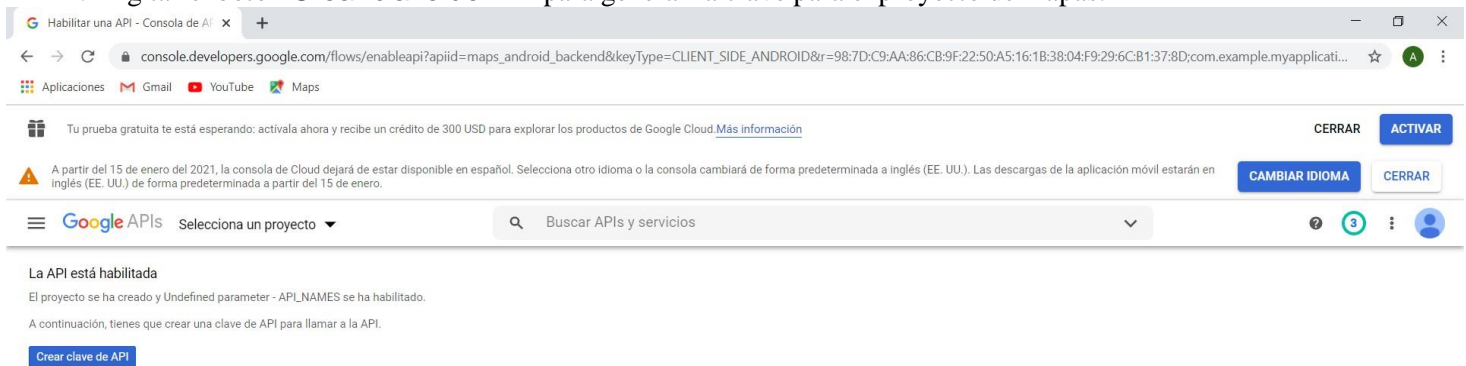


Figura 10. Digitar el botón Crear clave de API.

12. En la siguiente ventana Credenciales - APIs y servicios se muestra parte de la clave generada.

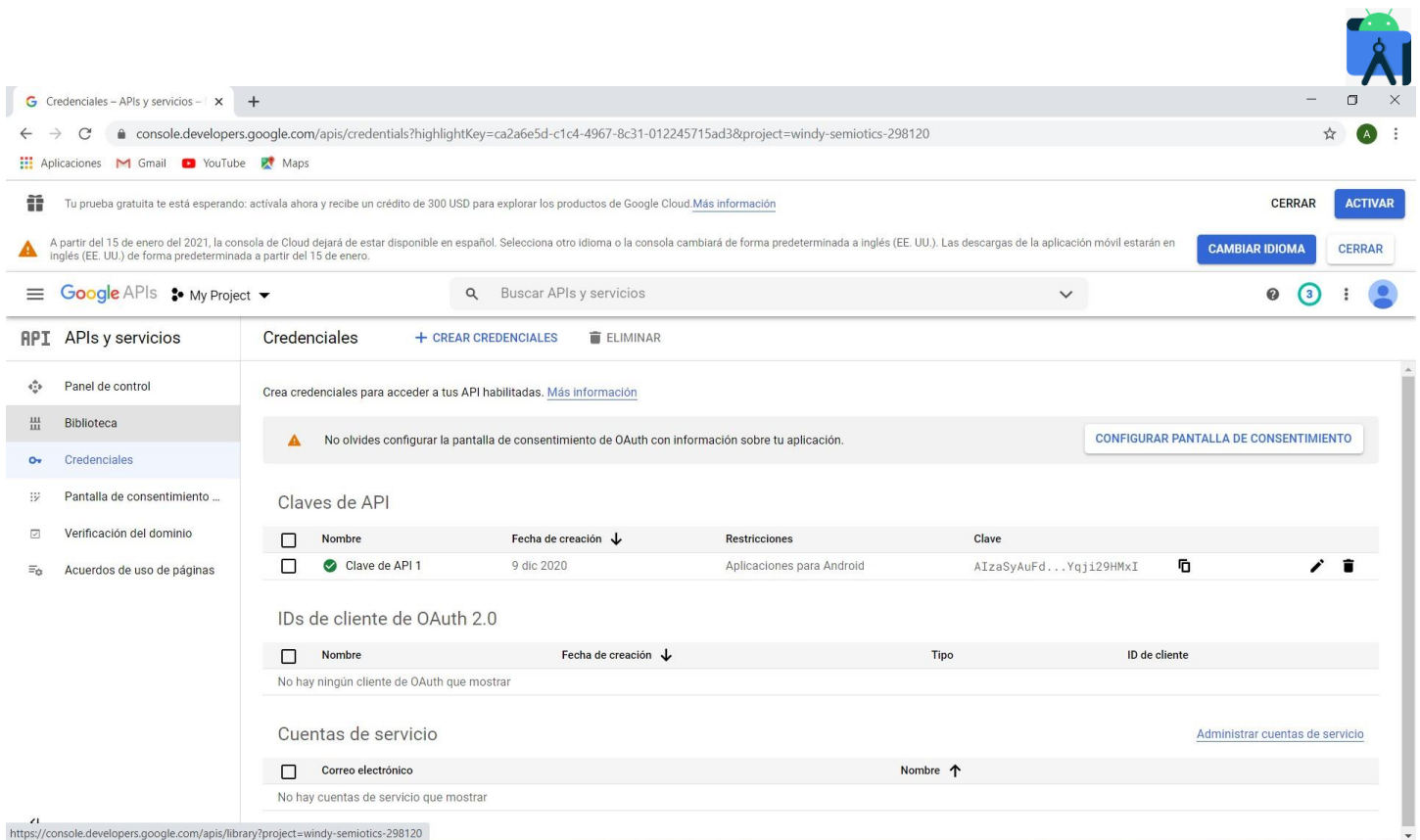


Figura 11. La clave generada inicia con AI z.

Digitar el botón  que se encuentra al lado de la clave. Copiar la clave y pegarla en el archivo correspondien

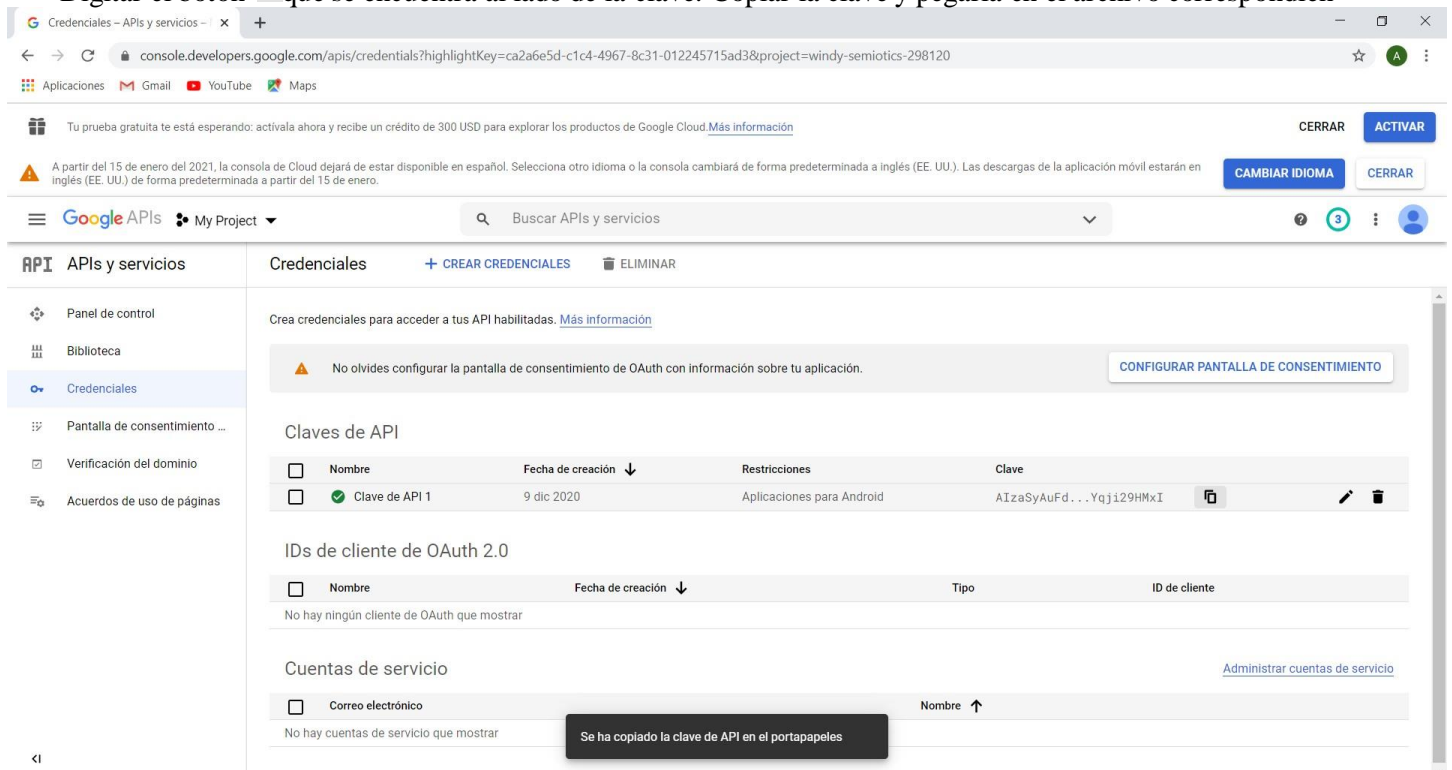



Figura 13. Copiar la clave generada con el botón .



13. Entre las etiquetas `<string>` y `</string>` se pegar la clave generada, como se indica en el siguiente código del archivo `google_maps_api.xml`.

```
<resources>
```

```
<!--
```

```
TODO: Before you run your application, you need a Google Maps API key.
```

```
To get one, follow this link, follow the directions and press "Create" at the end:
```

```
https://console.developers.google.com/flows/enableapi?apiid=maps\_android\_backend&keyType=CLIENT\_SIDE\_ANDROID&r=98:7D:C9:AA:86:CB:9F:22:50:A5:16:1B:38:04:F9:29:6C:B1:37:8D%3Bcom.example.myapplication
```

```
You can also add your credentials to an existing key, using these values:
```

```
Package name:
```

```
com.example.myapplication
```

```
SHA-1 certificate fingerprint:
```

```
98:7D:C9:AA:86:CB:9F:22:50:A5:16:1B:38:04:F9:29:6C:B1:37:8D
```

```
Alternatively, follow the directions here:
```

```
https://developers.google.com/maps/documentation/android/start#get-key
```

```
Once you have your key (it starts with "AIza"), replace the "google_maps_key" string in this file.
```

```
-->
```

```
<string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">
```

```
AIzaSyAuFdwKSwgsh4jCryAhGMqREYqji29HMxI
```

```
</string>
```

```
</resources>
```

14. En el archivo `AndroidManifest.xml` se observa la etiqueta de permisos `<uses-permission>` y la etiqueta `<meta-data>` con el atributo `android:value` con la cadena asignada a la clave en el archivo `strings.xml`.

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

```
<manifest xmlns:android="http://schemas.android.com/apk/res/android"  
package="com.example.myapplication">
```

```
<!--
```

```
The ACCESS_COARSE/FINE_LOCATION permissions are not required to use  
Google Maps Android API v2, but you must specify either coarse or fine  
location permissions for the "MyLocation" functionality.
```

```
-->
```

```
<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/Theme.MyApplication">
```

```
<!--
```



The API key for Google Maps-based APIs is defined as a string resource.



(See the file "res/values/google_maps_api.xml").

Note that the API key is linked to the encryption key used to sign the APK.

You need a different API key for each encryption key, including the release key that is used to

sign the APK for publishing.

You can define the keys for the debug and release targets in src/debug/ and src/release/.

-->

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

<activity
    android:name=".MapsActivity"
    android:label="@string/title_activity_maps">
    <intent-filter>
        <action android:name="android.intent.action.MAIN" />
        <category android:name="android.intent.category.LAUNCHER" />
    </intent-filter>
</activity>
</application>
</manifest>
```

15. El archivo strings.xml con el atributo name que indica la clave.

```
<resources>
    <string name="app_name">My Application</string>
    <string name="title_activity_maps">Map</string>
    <string name="google_maps_key">AIzaSyAuFdwKSwgsh4jCryAhGMqREYqji29HMxI</string>
</resources>
```

16. La ejecución del proyecto debe mostrar la imagen del mapa con las coordenadas predeterminadas de Sidney, Australia, como se indica en la siguiente figura. Esta aplicación ya se puede ejecutar en forma independiente sin pérdida de la localización.



Figura 14. La ejecución muestra el mapa de Australia.



NOTA: Capturar las imágenes, de la ejecución del ejercicio, en un documento y guardarlo con la sintaxis `AlumnoMapas1Grupo.pdf`. Enviar el archivo al sitio indicado por el profesor.