# Guía - Localización GPS y Api de mapas

## Paso 1: Obtener la API Key para el uso de Google Maps.

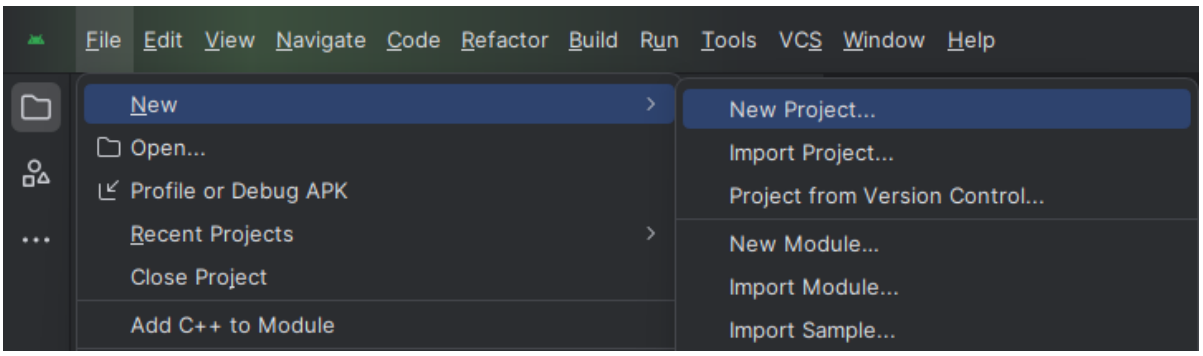
Para facilitar el proceso se entrega un API Key ya existente:

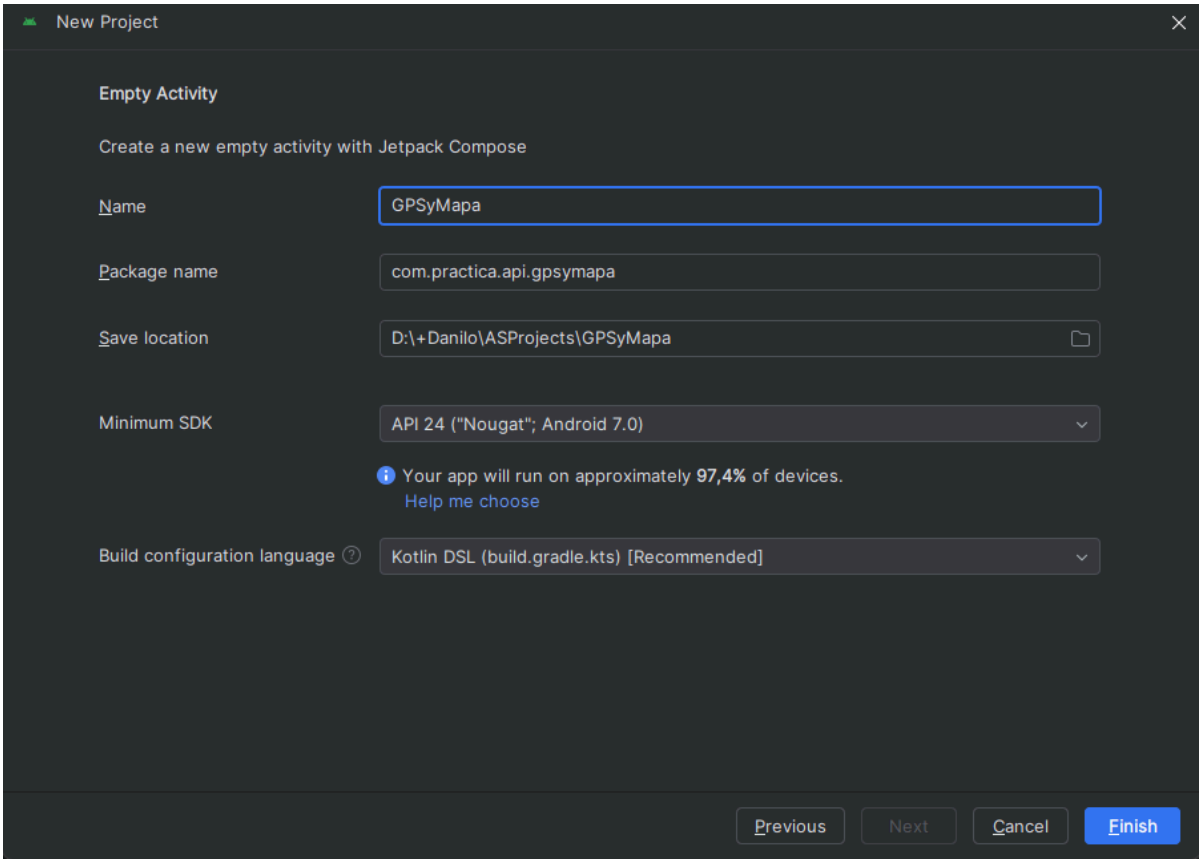
***API Key:***

Para más información sobre cómo obtener tu propia API Key ingresa aquí

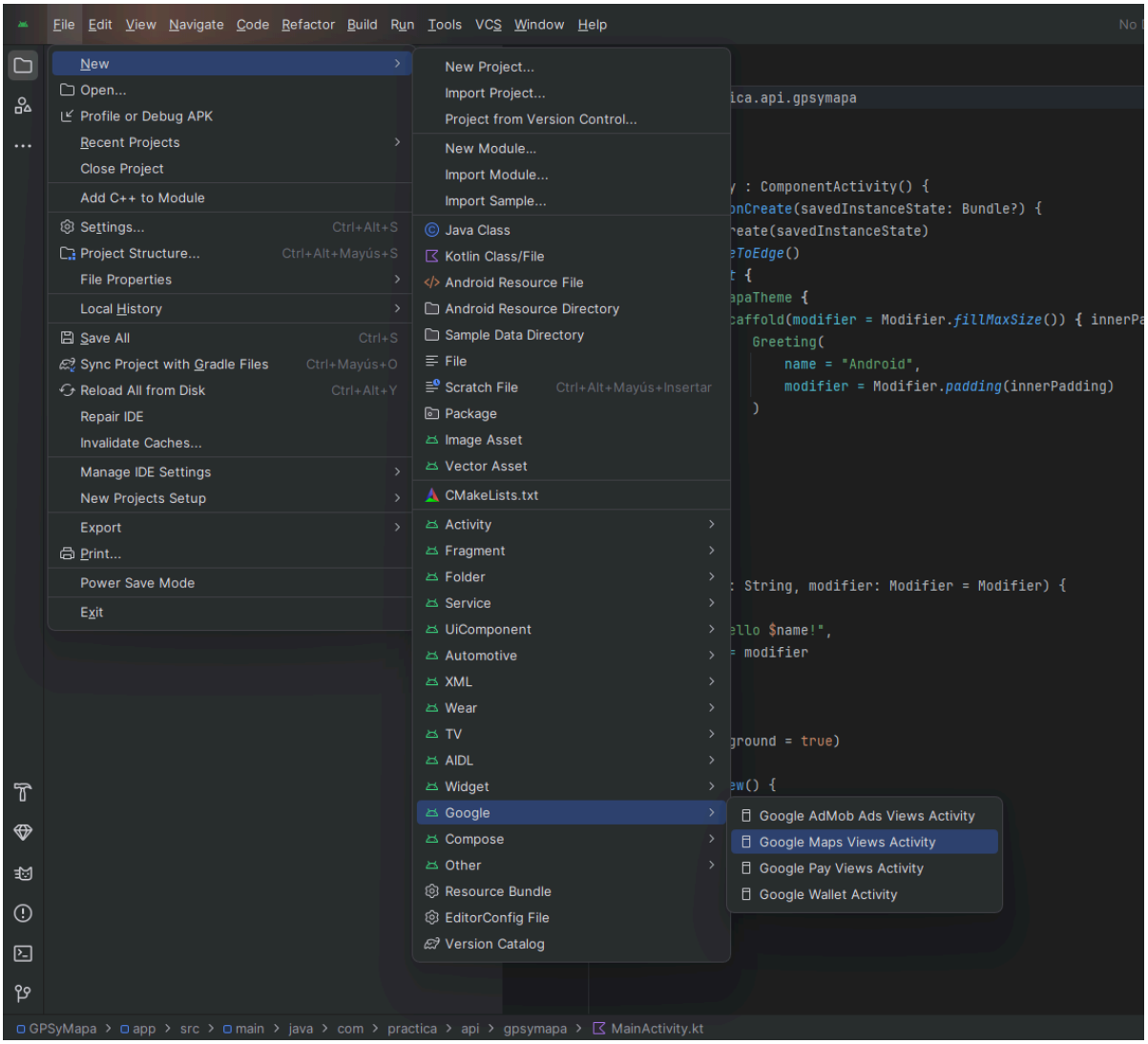
## Paso 2: Creación del proyectoi

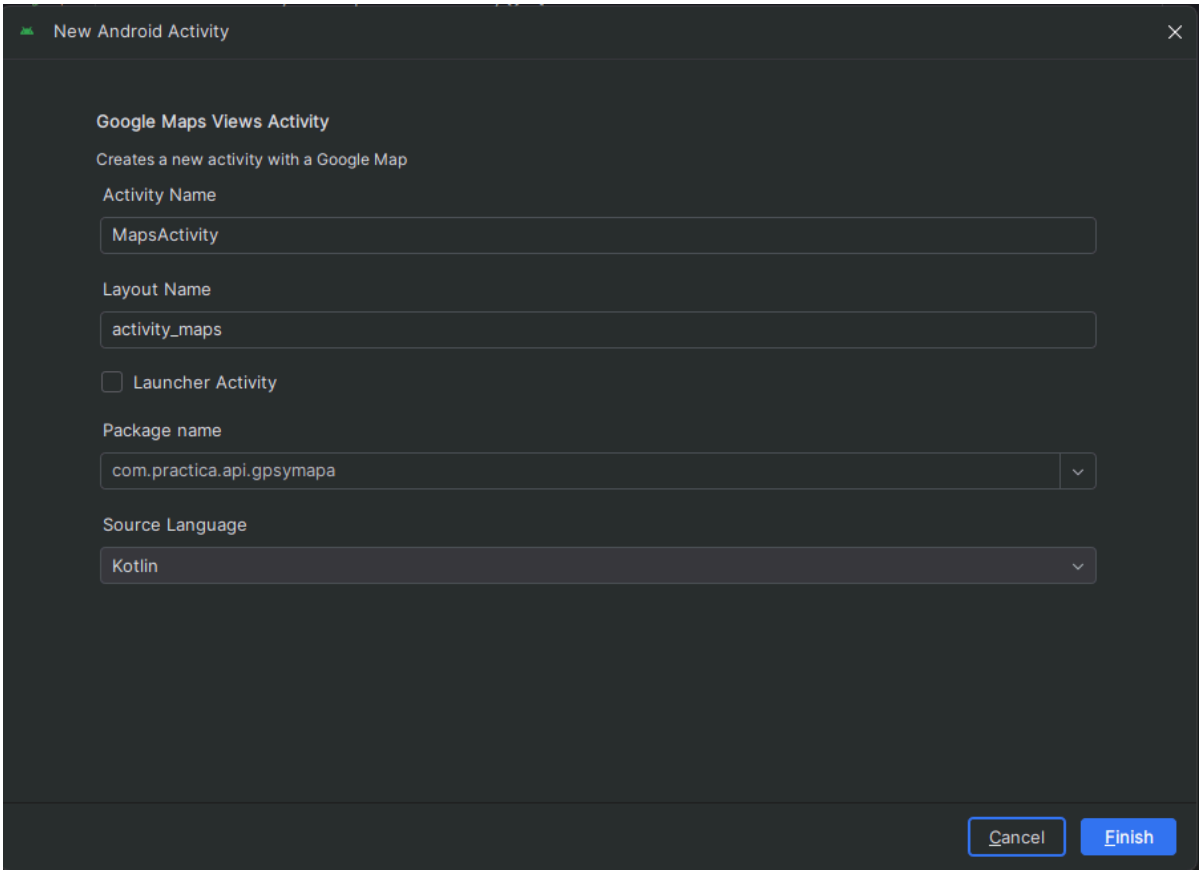
Creamos un nuevo proyecto, de tipo “Empty Activity”, y le damos un nombre adecuado:



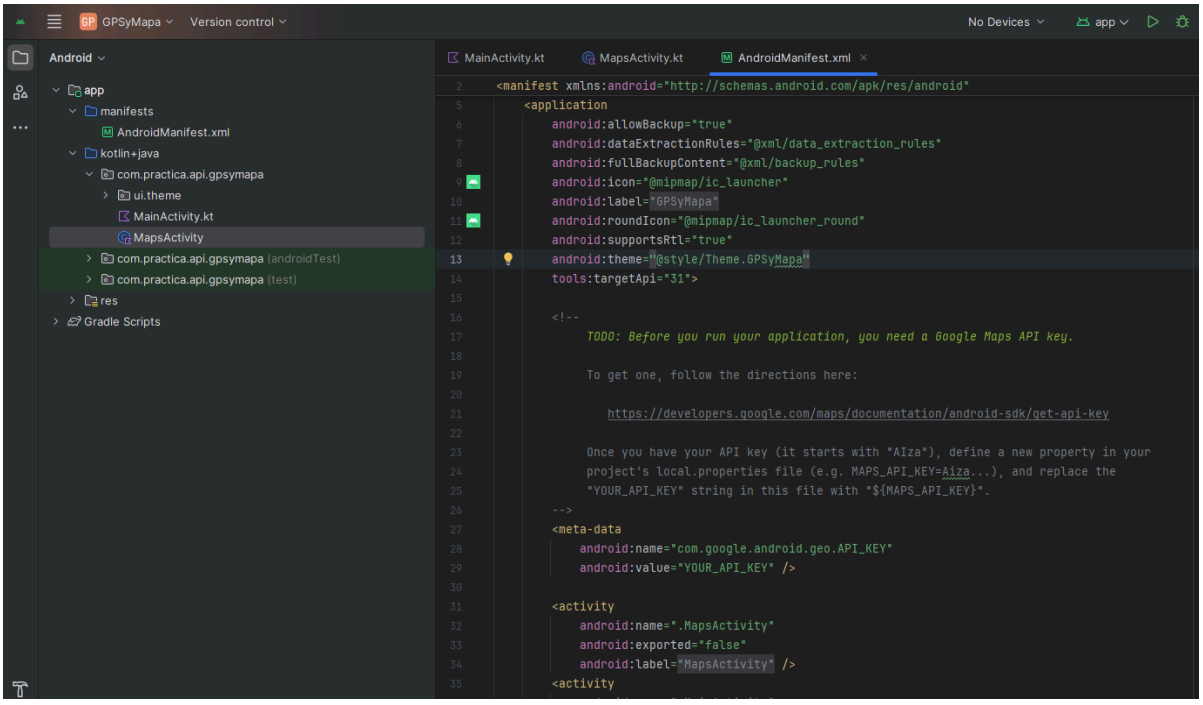




Una vez finalizada la configuración y construcción del proyecto, procedemos a crear una actividad de Google Maps inicial desde Android Studio:


File Edit View Navigate Code Refactor Build Run Tools VCS Window Help
New
Open...
Profile or Debug APK
Recent Projects
Close Project
Add C++ to Module
Settings... Ctrl+Alt+S
Project Structure... Ctrl+Alt+Mayús+S
File Properties
Local History
Save All Ctrl+S
Sync Project with Gradle Files Ctrl+Mayús+O
Reload All from Disk Ctrl+Alt+Y
Repair IDE
Invalidate Caches...
Manage IDE Settings
New Projects Setup
Export
Print...
Power Save Mode
Exit
New Project...
Import Project...
Project from Version Control...
New Module...
Import Module...
Import Sample...
Java Class
Kotlin Class/File
Android Resource File
Android Resource Directory
Sample Data Directory
File
Scratch File Ctrl+Alt+Mayús+Insertar
Package
Image Asset
Vector Asset
CMakeLists.txt
Activity
Fragment
Folder
Service
UiComponent
Automotive
XML
Wear
TV
AIDL
Widget
Google
Compose
Other
Resource Bundle
EditorConfig File
Version Catalog
Google AdMob Ads Views Activity
Google Maps Views Activity
Google Pay Views Activity
Google Wallet Activity
GPSyMapa › app › src › main › java › com › practica › api › gpsymapa › MainActivity.kt



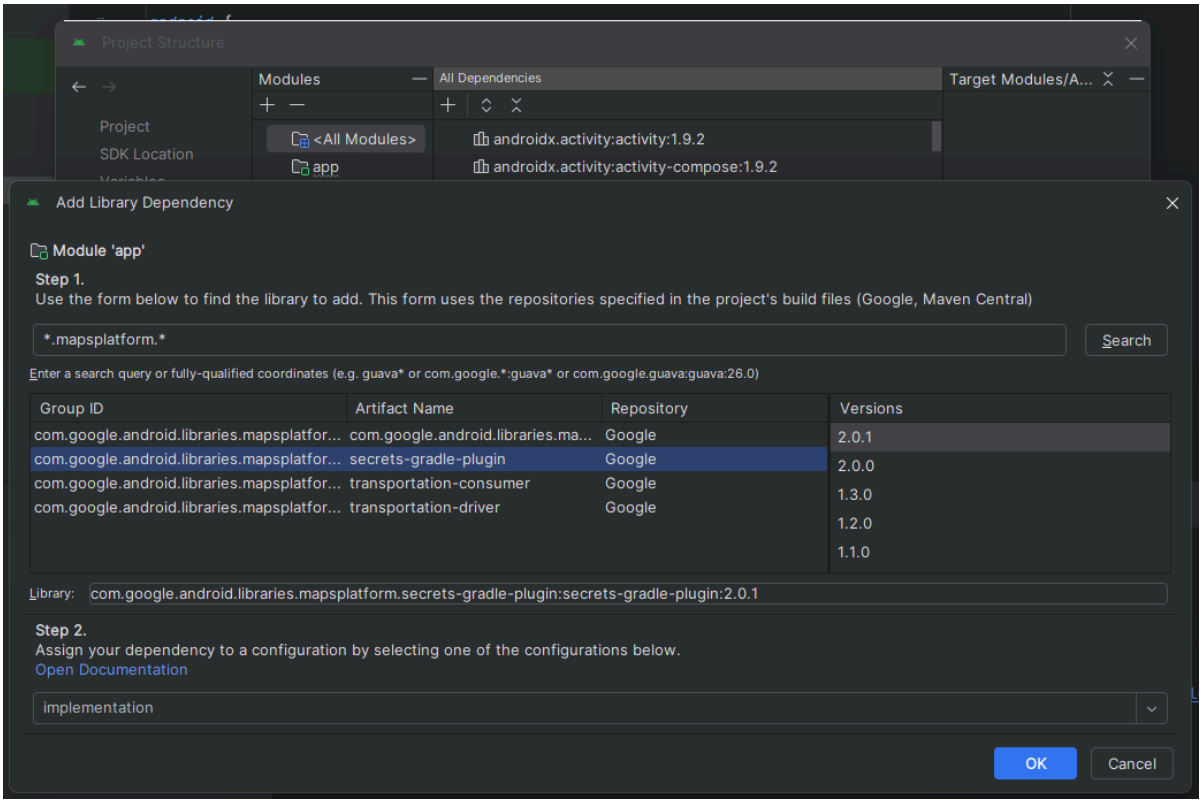

Al finalizar, observaremos que se ha creado una clase “MapsActivity” y se nos abrirá el AndroidManifest.xml.



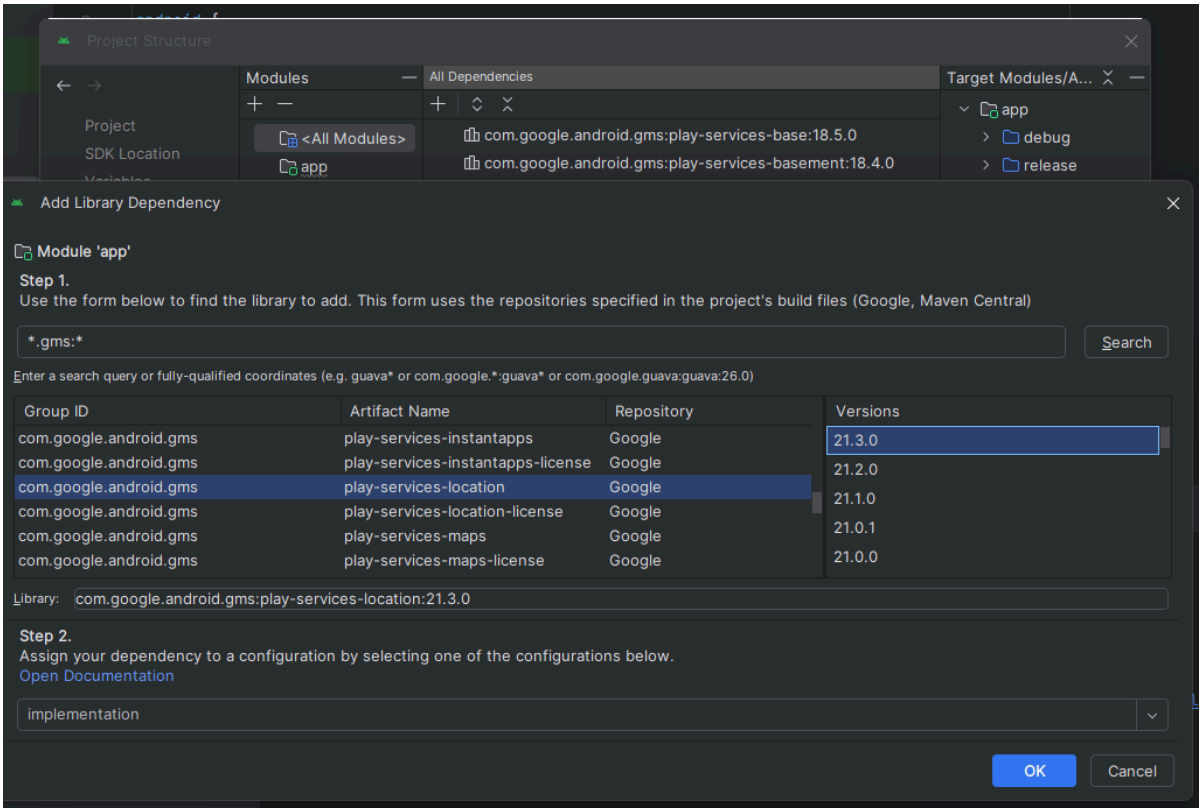
## Paso 3: Gestión de dependencias

Antes de continuar procedemos a gestionar las dependencias necesarias para el proyecto: Accedemos a Files, Project Structure, luego dependencies, y agregamos nuevas dependencias:

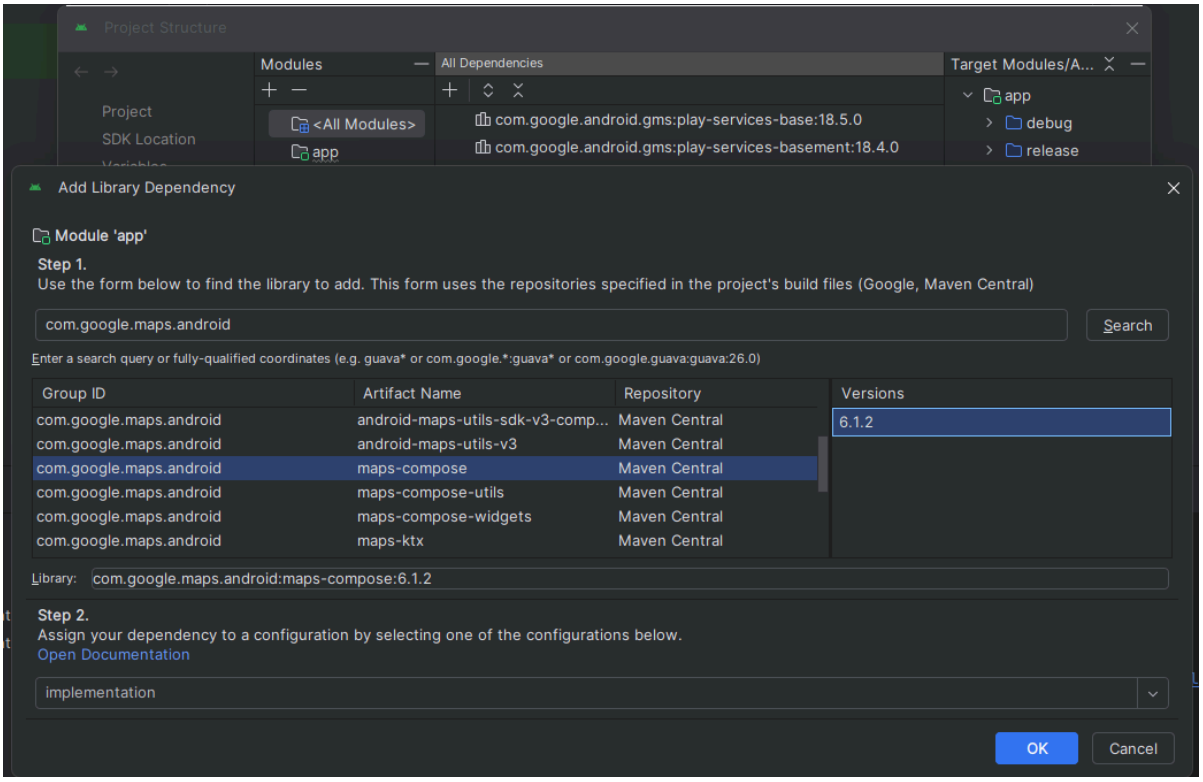
Primero secrets-gradle-plugin, buscando *.mapsplatform.*, como se muestra en la imagen:




Segundo, play-services-location, buscando *.gms:*, como se muestra en la imagenes:



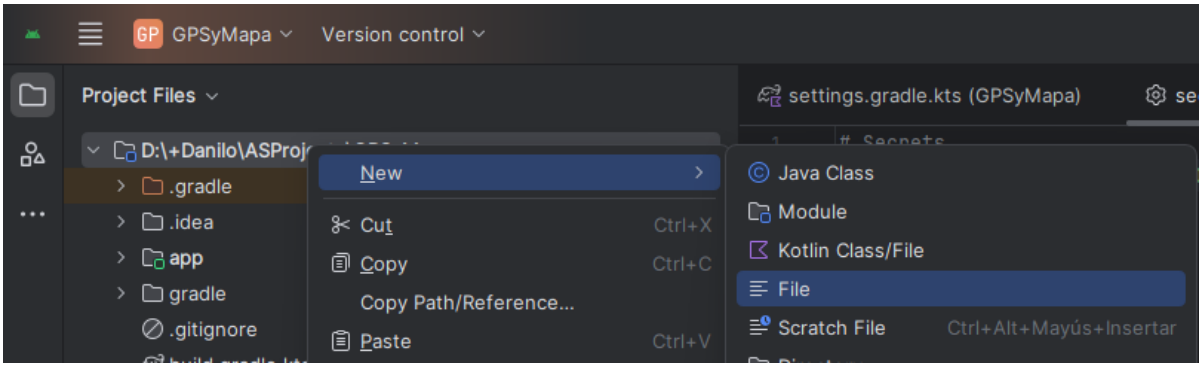

Tercero, agregar maps-compose, buscando com.google.maps.android, como se muestra en la imagen:

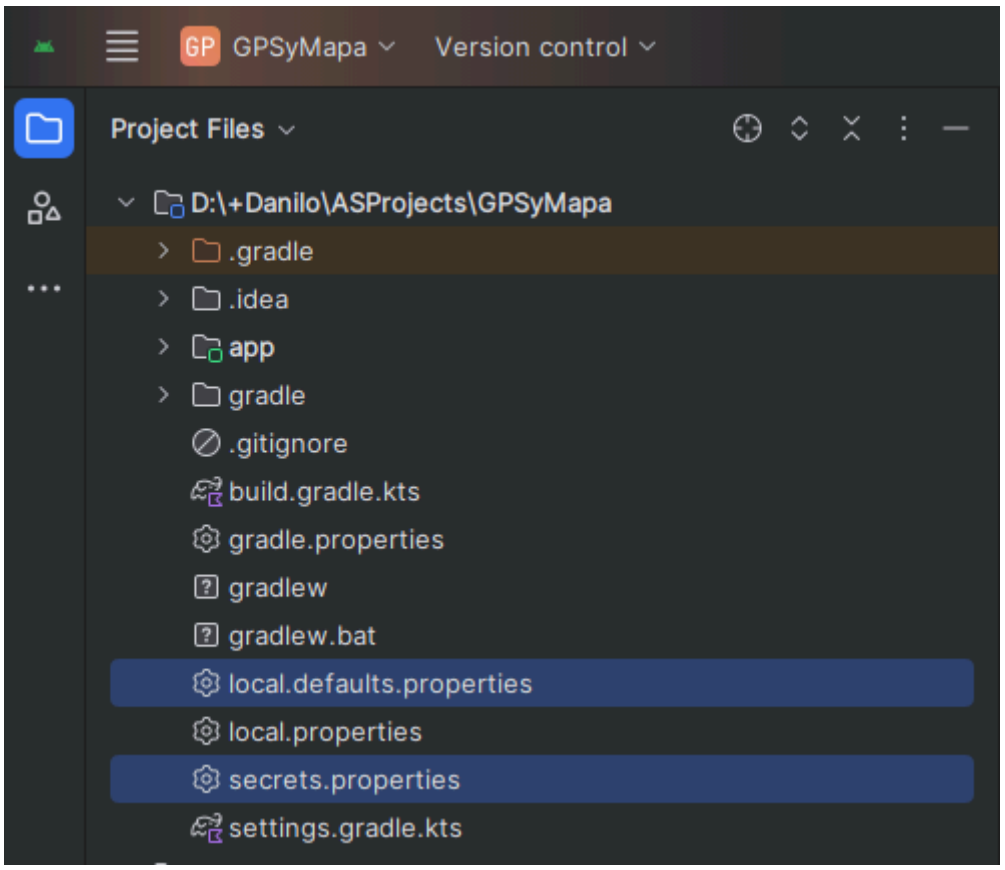



Una vez agregadas las dependencias, aplicamos los cambios y aceptamos. Y esperamos a que se sincronice el proyecto.

## Paso 4: Hacer uso de Gradle Secrets para la API Key

Para mantener la seguridad de nuestra API Key, crearemos 2 archivos properties en la sección Gradle Scripts, uno con el nombre “secrets.properties” y el otro con nombre “local.defaults.properties” (Recomendamos cambiar el modo de visualización a “Project Files” para la creación de los archivos):



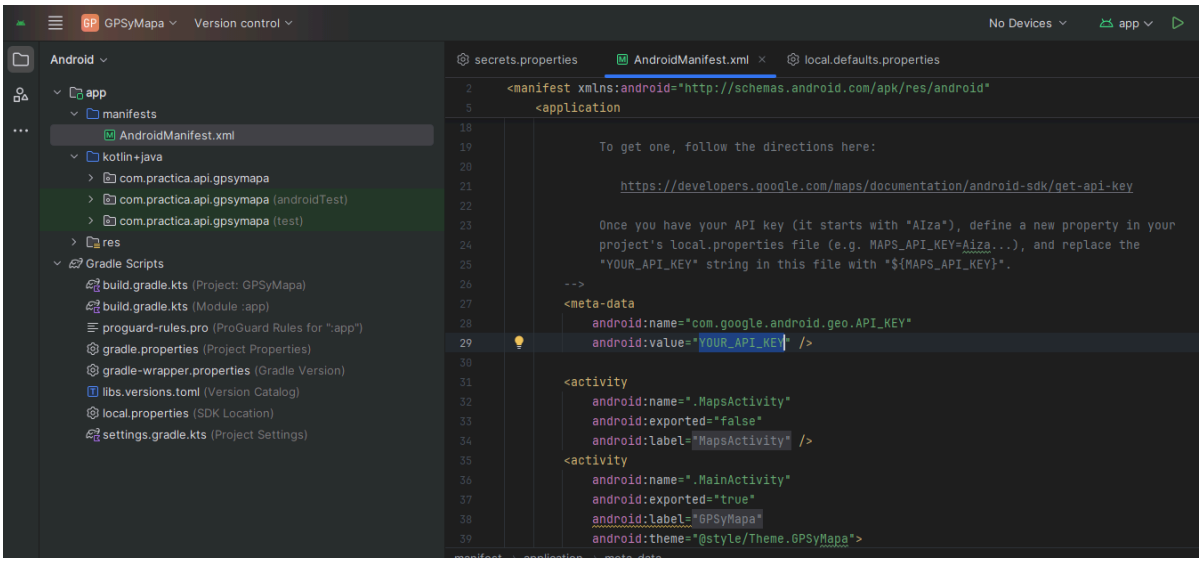

Luego, en el archivo secrets.properties escribimos lo siguiente, reemplazando el texto “API_KEY_A_UTILIZAR” por la llave que entregamos en el paso 1:

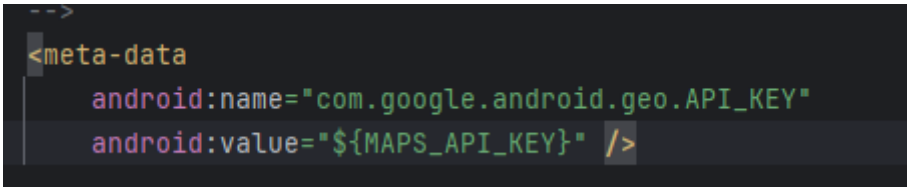
```
# Maps Api Secrets
MAPS_API_KEY=API_KEY_A_UTILIZAR
```

Y en el archivo local.defaults.properties la siguiente línea tal como está escrita:

```
MAPS_API_KEY=DEFAULT_API_KEY
```

Guardamos, y posteriormente regresamos a AndroidManifest.xml para reemplazar lo seleccionado por ${MAPS_API_KEY}:




Finalmente, ingresamos a build.gradle.kts (Module :app) y agregamos la siguiente sección al interior:

```
secrets {
    propertiesFileName = "secrets.properties"

    // A properties file containing default secret values. This
file can be
    // checked in version control.
    defaultPropertiesFileName = "local.defaults.properties"

    // Configure which keys should be ignored by the plugin by
providing regular expressions.
    // "sdk.dir" is ignored by default.
    ignoreList.add("keyToIgnore") // Ignore the key "keyToIgnore"
    ignoreList.add("sdk.*")       // Ignore all keys matching the
regexp "sdk.*"
}
```

## Paso 5: Localización GPS

Primero que todo agregamos los permisos necesarios para la geolocalizacion en el archivo de **AndroidManifest.xml** un permiso es para ubicación en primer plano y la otra es para segundo plano

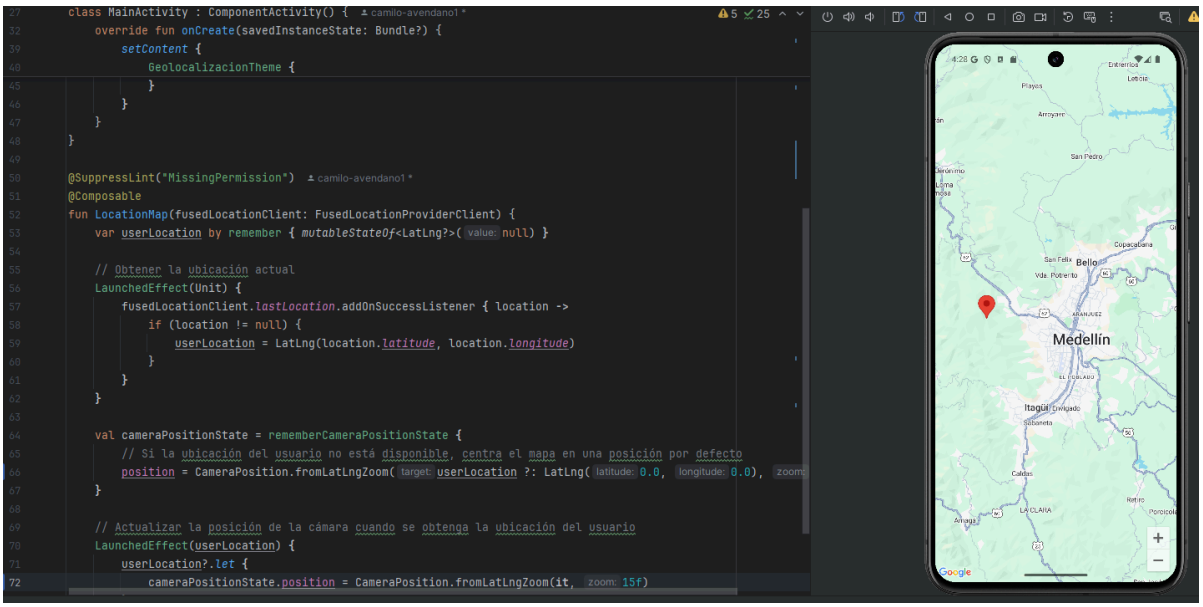
```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION"/>
<uses-permission android:name="android.permission.ACCESS_COARSE_LOCATION"/>
<uses-permission
android:name="android.permission.ACCESS_BACKGROUND_LOCATION"/>
```

## Paso 6: Google Maps

Una vez realizado lo anterior, podemos modificar el MainActivity.kt de la siguiente manera:

```
class MainActivity : ComponentActivity() {

   private lateinit var fusedLocationClient: FusedLocationProviderClient

   @SuppressLint("MissingPermission", "UnrememberedMutableState")
   override fun onCreate(savedInstanceState: Bundle?) {
       super.onCreate(savedInstanceState)

       // Inicializar el cliente de ubicación
       fusedLocationClient =
LocationServices.getFusedLocationProviderClient(this)

       enableEdgeToEdge()
       setContent {
           GeolocalizacionTheme {
               Surface(modifier = Modifier.fillMaxSize()) {

                   LocationMap(fusedLocationClient)
               }
           }
       }
   }
}

@SuppressLint("MissingPermission")
@Composable
fun LocationMap(fusedLocationClient: FusedLocationProviderClient) {
   var userLocation by remember { mutableStateOf<LatLng?>(null) }

   // Obtener la ubicación actual
   LaunchedEffect(Unit) {
       fusedLocationClient.lastLocation.addOnSuccessListener { location ->
           if (location != null) {
               userLocation = LatLng(location.latitude, location.longitude)
           }
       }
```

```
    }

    val cameraPositionState = rememberCameraPositionState {
        // Si la ubicación del usuario no está disponible, centra el mapa en
una posición por defecto
        position = CameraPosition.fromLatLngZoom(userLocation ?: LatLng(0.0,
0.0), 12f)
    }

    // Actualizar la posición de la cámara cuando se obtenga la ubicación del
usuario
    LaunchedEffect(userLocation) {
        userLocation?.let {
            cameraPositionState.position = CameraPosition.fromLatLngZoom(it,
15f)
        }
    }

    GoogleMap(
        modifier = Modifier.fillMaxSize(),
        cameraPositionState = cameraPositionState
    ) {
        userLocation?.let {
            // Coloca un marcador en la ubicación del usuario
            Marker(
                state = MarkerState(position = it),
                title = "Tu ubicación",
                snippet = "Estás aquí"
            )
        }
    }
}
```



Luego podemos hacer uso del manejo de permisosde la siguiente manera

```
class MainActivity : ComponentActivity() {

   private lateinit var fusedLocationClient: FusedLocationProviderClient

   // Registrar el contrato para la solicitud de permisos
   private val requestLocationPermissionLauncher = registerForActivityResult(
       ActivityResultContracts.RequestPermission()
   ) { isGranted: Boolean ->
       if (isGranted) {
           // Si los permisos fueron otorgados, obtener la ubicación actual
           setContent {
               GeolocalizacionTheme {
                   Surface(modifier = Modifier.fillMaxSize()) {
                       LocationMap(fusedLocationClient)
                   }
               }
           }
       } else {
           // Si los permisos fueron denegados, mostrar un mensaje
           Toast.makeText(this, "Permisos de ubicación denegados",
Toast.LENGTH_LONG).show()
           finish() // Finaliza la aplicación si no se conceden los permisos
       }
   }

   @SuppressLint("MissingPermission")
   override fun onCreate(savedInstanceState: Bundle?) {
       super.onCreate(savedInstanceState)

       // Inicializar el cliente de ubicación
       fusedLocationClient =
LocationServices.getFusedLocationProviderClient(this)

       // Verificar si los permisos están otorgados
       if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
           // Si los permisos no están otorgados, solicitarlos

requestLocationPermissionLauncher.launch(Manifest.permission.ACCESS_FINE_LOC
ATION)
       } else {
           // Si los permisos ya están otorgados, mostrar el mapa
           setContent {
               GeolocalizacionTheme {
                   Surface(modifier = Modifier.fillMaxSize()) {
                       LocationMap(fusedLocationClient)
                   }
               }
           }
```

```
        }
    }
}

@SuppressLint("MissingPermission")
@Composable
fun LocationMap(fusedLocationClient: FusedLocationProviderClient) {
    var userLocation by remember { mutableStateOf<LatLng?>(null) }

    // Obtener la ubicación actual
    LaunchedEffect(Unit) {
        fusedLocationClient.lastLocation.addOnSuccessListener { location ->
            if (location != null) {
                userLocation = LatLng(location.latitude, location.longitude)
            }
        }
    }

    val cameraPositionState = rememberCameraPositionState {
        // Si la ubicación del usuario no está disponible, centra el mapa en
una posición por defecto
        position = CameraPosition.fromLatLngZoom(userLocation ?: LatLng(0.0,
0.0), 11f)
    }

    // Actualizar la posición de la cámara cuando se obtenga la ubicación del
usuario
    LaunchedEffect(userLocation) {
        userLocation?.let {
            cameraPositionState.position = CameraPosition.fromLatLngZoom(it,
11f)
        }
    }

    GoogleMap(
        modifier = Modifier.fillMaxSize(),
        cameraPositionState = cameraPositionState
    ) {
        userLocation?.let {
            // Coloca un marcador en la ubicación del usuario
            Marker(
                state = MarkerState(position = it),
                title = "Tu ubicación",
                snippet = "Estás aquí"
            )
        }
    }
}
```

Referencias:

- Guía de inicio rápido del SDK de Maps para Android | Maps SDK for Android | Google for Developers
- Obtener ubicación GPS con Kotlin y Android Studio (youtube.com)
- Biblioteca de Maps Compose | Maps SDK for Android | Google for Developers