

# GUIA 24 - Mapas en Android

## Objetivo

En esta guía aprenderemos a utilizar los Servicios de Mapas que existen en Android, para que a futuro puedas implementarlo en tus aplicaciones.

## Mapas en Android

Para integrar funcionalidades de mapas en nuestras apps, tenemos varias opciones a nuestro alcance. Las opciones más usadas en Android son el API de Google Maps y otras SDK Basadas en Open Street Maps como MapBox, aunque también se tiene disponible MapKit de Apple.

Según muchos desarrolladores consideran que la opción más recomendada para el uso de mapas en nuestras apps, es el uso de Mapbox, por su simplicidad y flexibilidad (es 100% personalizable y de código abierto inclusive posee más funcionalidades que otros frameworks no ofrecen).

En esta guía aprenderemos a integrar Google Maps y Mapbox en nuestras aplicaciones.

## Google Maps

El SDK de Google Maps para android permite aprovechar todas las ventajas que ofrece la solución Google Maps dentro de nuestras apps, permite agregar marcadores, polígonos y superposiciones. Además del soporte de extensiones y funciones avanzadas.

## Pasos para integrar Google Maps

El proceso general para añadir Google Maps en nuestro proyecto consiste en:

1. Obtener una Clave API
2. Agregar configuraciones en el archivo de manifiesto
3. Agregar la dependencia de Gmaps en el build.gradle
4. Agregar MapView

## Obtener una Clave API

Para obtener una clave API de Google Maps debemos acceder con nuestro correo de gmail a Google Cloud Console <https://console.cloud.google.com/>. Una vez que nos encontremos en el panel principal de Google Cloud Console debemos seleccionar un proyecto existente o crear uno nuevo:

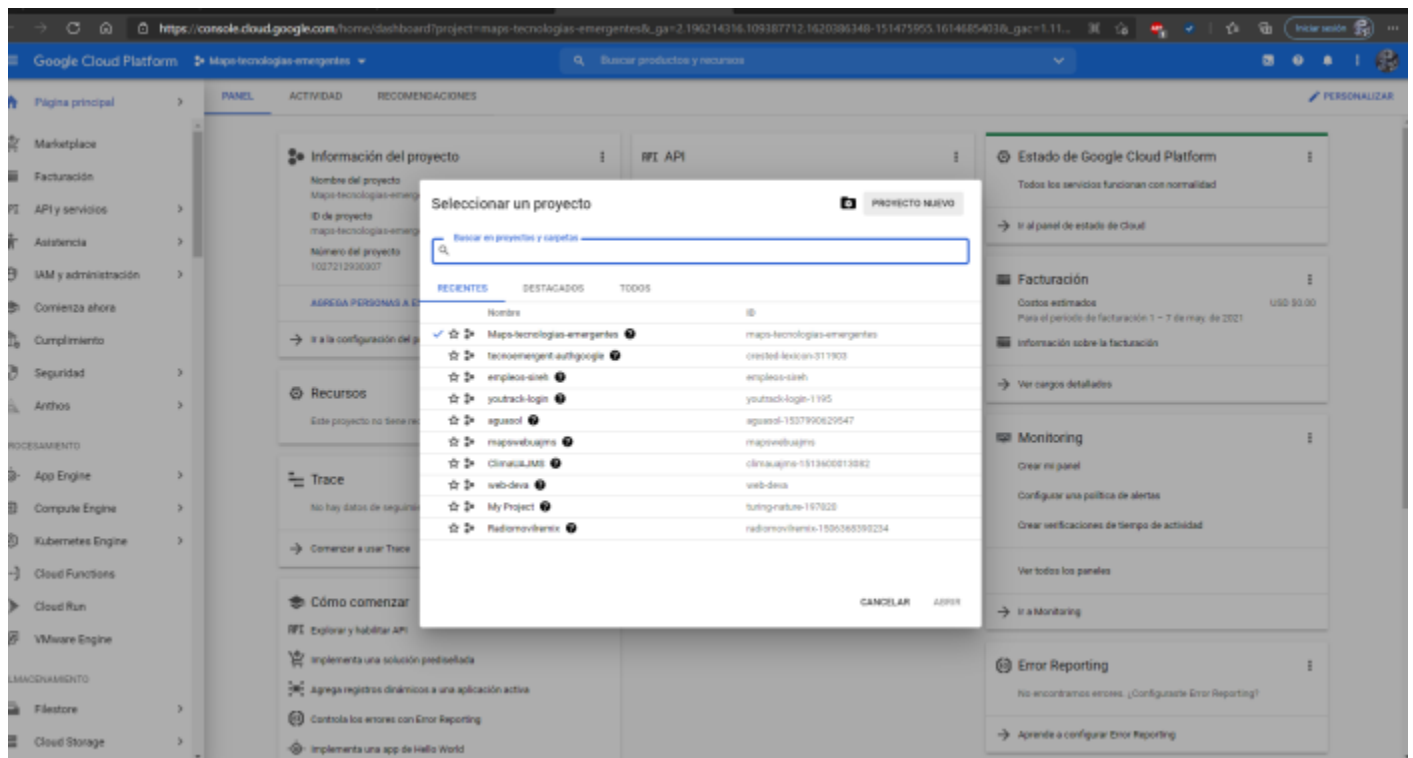


Figure 1: Pantalla de creación de proyectos de Google Cloud Console

Cuando seleccionamos un proyecto de Google podremos acceder a muchas otras funcionalidades relacionadas a nuestro proyecto. Si hacemos clic en *Ir a la descripción general de las API*, nos llevará a la sección para configurar las API KEY

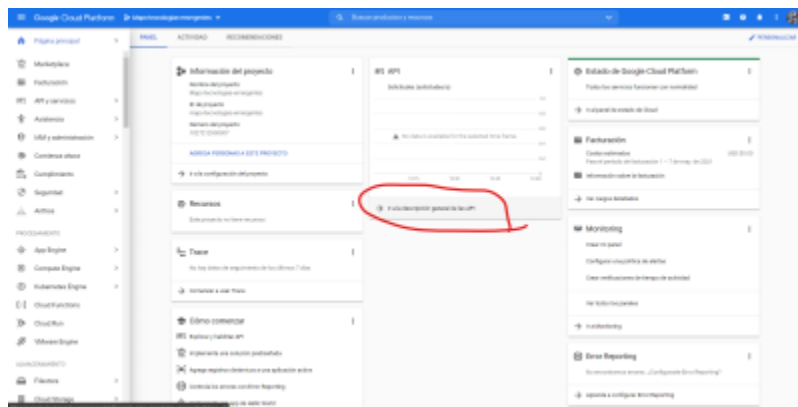


Figure 2: Pantalla principal del Proyecto Google

Una vez que no encontremos en la sección de API y Servicios seleccionamos credenciales:

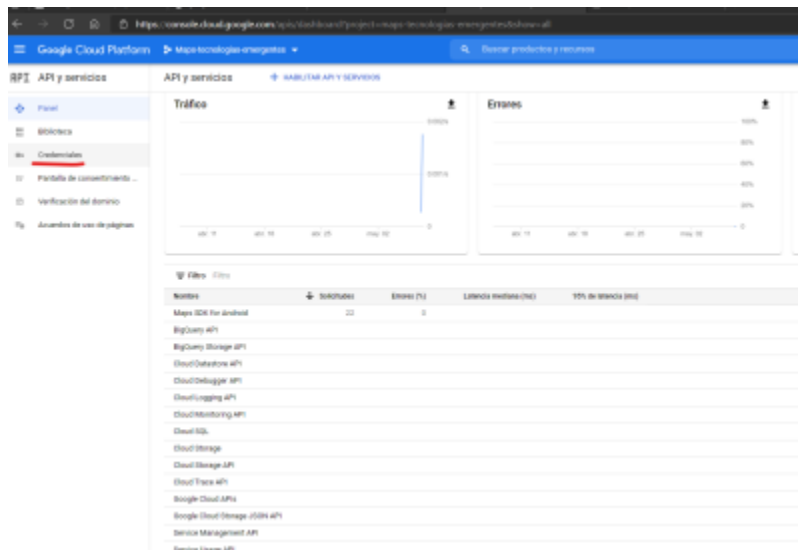


Figure 3: Pantalla de API y Servicio de Proyecto Google

Desde la pantalla principal de credenciales podremos crear nuevas credenciales o usar las existentes:

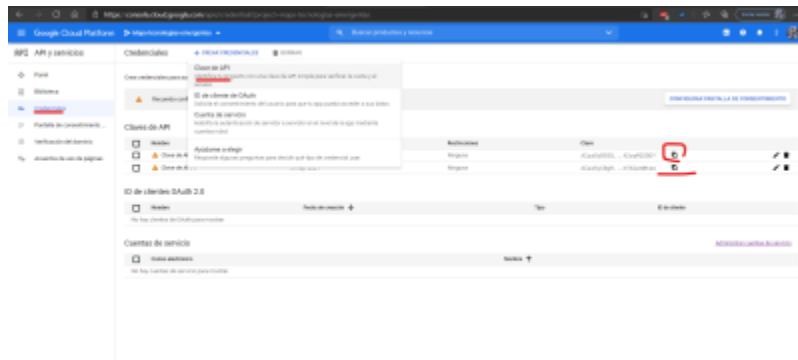


Figure 4: Pantalla de administración de Credenciales

Haciendo clic sobre una de las claves, el sistema nos llevará a una sección donde podremos configurar las restricciones respecto al uso de nuestra clave de API seleccionada, desde la cual podremos copiar la Clave de API:

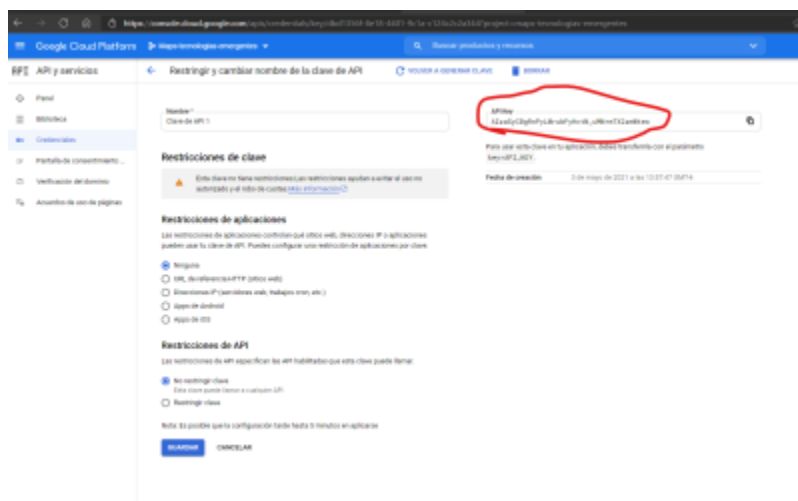


Figure 5: Pantalla de configuración de Clave

## Agregar configuraciones en el archivo de manifiesto

Una vez que tenemos la clave de API debemos agregar ésta al archivo de manifiesto:

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

Además debemos añadir los permisos necesarios en el mismo archivo:

```
<manifest>
...
    <uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
    <uses-permission android:name="android.permission.INTERNET"/>
    <uses-permission android:name="android.permission.ACCESS_NETWORK_STATE"/>
...
</manifest>
```

## Agregar la dependencia de Gmaps en el build.gradle

Para agregar la dependencia del SDK de Google Maps, es necesario añadir la siguiente dependencia en el build.gradle del módulo:

```
implementation 'com.google.android.gms:play-services-maps:18.2.0'//para mapas
```

Adicionalmente si es que vamos hacer uso de la geolocalización podemos agregar la dependencia play-services-location:

```
implementation 'com.google.android.gms:play-services-location:21.2.0'//para
localizacion
```

## Agregar un Mapa

Para agregar un Mapa de Google en una de nuestras actividades podemos usar Mapview o usar un Fragment del tipo com.google.android.gms.maps.SupportMapFragment. En nuestro ejemplo usaremos el componente MapView. Para ello añadimos en nuestro layout principal el componente com.google.android.gms.maps.MapView:

```
<com.google.android.gms.maps.MapView
    android:id="@+id/mapView"
    android:layout_width="match_parent"
    android:layout_height="match_parent">
```

`/>`

En el archivo de contexto agregamos la siguiente linea despues de ejecutar el método `setContentView()` dentro del método `onCreate()`:

```
...
private val MAPVIEW_BUNDLE_KEY = "MapViewBundleKey"
....
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);

    // *** IMPORTANT ***
    // MapView requires that the Bundle you pass contain _ONLY_ MapView SDK
    // objects or sub-Bundles.
    Bundle mapViewBundle = null;
    var mapViewBundle: Bundle? = null
    if (savedInstanceState != null) {
        mapViewBundle = savedInstanceState.getBundle(MAPVIEW_BUNDLE_KEY)
    }
    mMapView = findViewById(R.id.mapView);
    mMapView.onCreate(mapViewBundle);
    mMapView.getMapAsync(this);
}
```

Cambiamos la actividad principal para que ésta implemente la interfaz `com.google.android.libraries.maps.OnMapReadyCallback`, el cual tiene un método `onMapReady()`, que se ejecuta cuando el mapa se carga correctamente.

```
class MainActivity : AppCompatActivity(), OnMapReadyCallback {

    ...
    var myMap: GoogleMap? = null
    var lt = -21.5360986
    var lg = -64.7154756
    override fun onMapReady(map: GoogleMap) {
        myMap = map
        val markerOptions1 =
            MarkerOptions().position(LatLng(-21.585975, -64.700308)).title("Mi
Ubicacion1")

        map.addMarker(markerOptions1)

        val markerOptions2 =
            MarkerOptions().position(LatLng(lt, lg)).title("Mi Ubicacion 2")

        map.addMarker(markerOptions2)
```

```
val uiSettings: UiSettings = map.getUiSettings()
uiSettings.isZoomControlsEnabled = true // Habilitar controles de zoom

val Liberty =
    CameraPosition.builder().target(LatLng(lt,
lg)).zoom(16f).bearing(0f).tilt(45f).build()
    map.moveCamera(CameraUpdateFactory.newCameraPosition(Liberty))

}
...
}
```

Finalmente dentro del método onCreate inicializamos el mapa de la siguiente forma:

```
override fun onCreate(savedInstanceState: Bundle?) {
    super.onCreate(savedInstanceState)
    setContentView(R.layout.activity_main);

    // *** IMPORTANT ***
    // MapView requires that the Bundle you pass contain _ONLY_ MapView SDK
    // objects or sub-Bundles.
    var mapViewBundle: Bundle? = null
    if (savedInstanceState != null) {
        mapViewBundle = savedInstanceState.getBundle(MAPVIEW_BUNDLE_KEY)
    }
    mMapView = findViewById(R.id.mapView)
    mMapView.onCreate(mapViewBundle)
    mMapView.getMapAsync(this)
}

override fun onLowMemory() {
    super.onLowMemory()
    mMapView.onLowMemory();
}

override fun onPause() {
    mMapView.onPause()
    super.onPause()
}

override fun onDestroy() {
    mMapView.onDestroy()
    super.onDestroy()
}

override fun onResume() {
    super.onResume()
}
```

```
mMapView.onResume()  
}  
  
override fun onStart() {  
    super.onStart()  
    mMapView.onStart()  
}  
  
override fun onStop() {  
    super.onStop()  
    mMapView.onStop()  
}
```

## Añadir un marcador

Para añadir un marcador debemos hacerlo dentro del método `onMapReady()`:

```
var myMap: GoogleMap? = null  
var lt = -21.5360986  
var lg = -64.7154756  
  
override fun onMapReady(GoogleMap map) {  
    myMap = map  
    val markerOptions1 =  
        MarkerOptions().position(LatLng(-21.585975, -64.700308)).title("Mi  
Ubicacion1")  
  
    map.addMarker(markerOptions1)  
}
```



Figure 6: App de Google Maps en funcionamiento

Link del repositorio: <https://github.com/F1852D160/MapsEjemplo.git>

## Mapbox

Mapbox surge como alternativa fuerte a Google Maps debido a que desde el año 2018 Google introdujo una serie de



limitaciones de uso en el API Google Maps. Por tal motivo hace que el acceso a este servicio sea algo caro.

## Pasos para integrar Mapbox

Para integrar el SDK de Mapbox en nuestro proyecto primeramente debemos crear una cuenta en mapbox.com y seguir los pasos indicados en la guía oficial <https://docs.mapbox.com/android/maps/guides/install/>, que se resumen en lo siguiente:

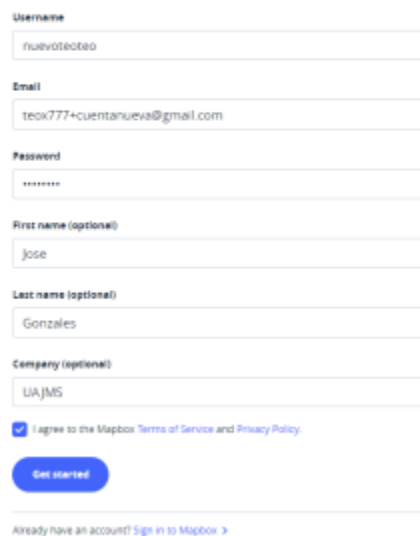
1. Crear una cuenta en mapbox
2. Generar un token de acceso
3. Añadir el token a nuestro proyecto
4. Añadir la dependencia del SDK
5. Añadir un Mapa

## Crear una cuenta en mapbox

Para crear una cuenta debemos ingresar a: <https://account.mapbox.com/auth/signup/>, registrarnos en el formulario y aceptar los términos y condiciones de uso de Mapbox.

### Create your Mapbox account

Start using our world-class design and development tools for apps, navigation, AR, and data visualization. Already have an account? [Log in](#).



The screenshot shows the 'Create your Mapbox account' form. It includes fields for Username (nuevotesteo), Email (teox777+cuentanueva@gmail.com), Password (masked with dots), First name (optional) (Jose), Last name (optional) (Gonzales), and Company (optional) (UAJMS). There is a checkbox for 'I agree to the Mapbox Terms of Service and Privacy Policy' which is checked. A blue 'Get started' button is at the bottom. A link 'Already have an account? Sign in to Mapbox >' is at the very bottom.

Figure 7: Formulario de registro de cuenta en Mapbox

Una vez que confirmemos la creación desde nuestra de correo proporcionado, ya podremos iniciar sesión en Mapbox.

## Generar un token de acceso

Para generar un token de acceso debemos iniciar sesión y acceder a la sección de creación de tokens

<https://account.mapbox.com/access-tokens/create>

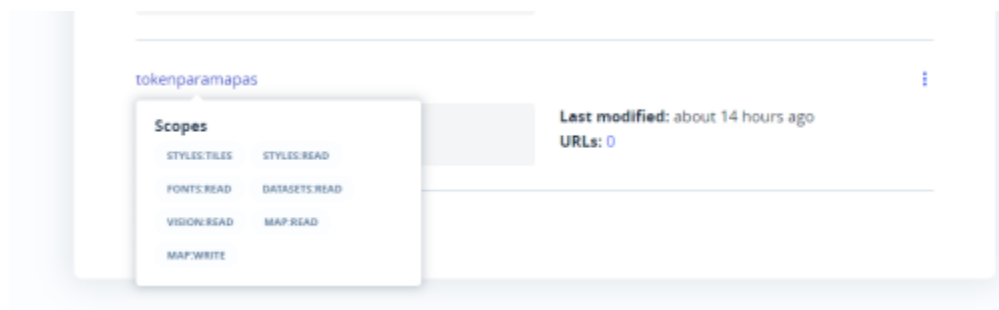


Figure 8: Permisos de un token de mapbox

Un token para que tenga los permisos de cargar mapas debe tener habilitado por lo menos el permiso MAP:READ, tal como se muestra en la figura.

## Añadir el token a nuestro proyecto

El token generado en mapbox nos servira tanto para descargar las librerias del SDK, como tambien para visualizar los mapas, por tanto se debe registrar en /res/values/strings.xml el siguiente item:

```
<string name="mapbox_access_token">TU_TOKEN_DE_MAPBOX</string>
```

Debes añadir el permiso para acceder a la ubicación en el archivo de manifiesto:

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

## Añadir la dependencia del SDK

Debes añadir en el build.gradle del módulo la dependencia del SDK

```
dependencies {  
    ...  
    implementation 'com.mapbox.mapboxsdk:mapbox-android-sdk:9.6.1'  
    ...  
}
```

En el build.gradle del proyecto en la sección allprojects, debes añadir lo siguiente:

```
allprojects {  
    repositories {  
        ...  
        maven {  
            url 'https://api.mapbox.com/downloads/v2/releases/maven'  
            authentication {  
                basic(BasicAuthentication)  
            }  
            credentials {  
                // Do not change the username below.  
            }  
        }  
    }  
}
```

```
// This should always be `mapbox` (not your username).
username = 'mapbox'
// Use the secret token you stored in gradle.properties as the
password
password = "TU_TOKEN_DE_MAPBOX"
    }
}
...
}
```

Una vez realizado los cambios en el archivo build.gradle no debemos olvidar sincronizar la configuración de nuestro proyecto.

## Añadir un Mapa

Para añadir un mapa en el archivo de la vista debemos agregar el componente `com.mapbox.mapboxsdk.maps.MapView`:

```
<com.mapbox.mapboxsdk.maps.MapView
    android:id="@+id/mapView"
    android:layout_width="match_parent"
    android:layout_height="match_parent"
/>
```

En el archivo de Contexto de nuestra actividad instanciamos el mapa de la siguiente forma:

```
private MapView mapView;

@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);

    Mapbox.getInstance(this, getString(R.string.mapbox_access_token));

    setContentView(R.layout.activity_main);

    mapView = (MapView) findViewById(R.id.mapView);
    mapView.onCreate(savedInstanceState);
    mapView.getMapAsync(new OnMapReadyCallback() {
        @Override
        public void onMapReady(@NonNull MapboxMap mapboxMap) {

            mapboxMap.setStyle(Style.MAPBOX_STREETS, new Style.OnStyleLoaded() {
                @Override
                public void onStyleLoaded(@NonNull Style style) {

                    // Map is set up and the style has loaded. Now you can add data or make
                    other map adjustments
                }
            })
        }
    })
}
```

```
    }  
    });  
}  
});  
}
```

El componente MapView de Mapbox maneja internamente su ciclo de vida por lo tanto debemos sobrescribir los métodos onStart(), onResume(), onPause(), onStop(), onSaveInstanceState(), onLowMemory(), onDestroy() de la siguiente forma:

```
@Override  
protected void onStart() {  
    super.onStart();  
    mapView.onStart();  
}  
  
@Override  
protected void onResume() {  
    super.onResume();  
    mapView.onResume();  
}  
  
@Override  
protected void onPause() {  
    super.onPause();  
    mapView.onPause();  
}  
  
@Override  
protected void onStop() {  
    super.onStop();  
    mapView.onStop();  
}  
  
@Override  
protected void onSaveInstanceState(Bundle outState) {  
    super.onSaveInstanceState(outState);  
    mapView.onSaveInstanceState(outState);  
}  
  
@Override  
public void onLowMemory() {  
    super.onLowMemory();  
    mapView.onLowMemory();  
}  
  
@Override  
protected void onDestroy() {  
    super.onDestroy();  
    mapView.onDestroy();  
}
```

```
}
```

## Cambiar el Estilo de mapa

Dentro del Método `onMapReady()`, se puede comenzar a personalizar la forma de como se visualizará nuestro mapa, una de las primeras opciones es seleccionar el estilo del mapa. Los incluidos por defecto son: `Style.MAPBOX_STREETS`, `Style.OUTDOORS`, `Style.LIGHT`, `Style.DARK`, `Style.SATELLITE`, `Style.SATELLITE_STREETS`, `Style.TRAFFIC_DAY`, `Style.TRAFFIC_NIGHT`.

```
mapboxMap.setStyle(Style.MAPBOX_STREETS, new Style.OnStyleLoaded() {  
    @Override  
    public void onStyleLoaded(@NonNull Style style) {  
  
    }  
});
```

Aunque también podemos definir nuestros propios estilos desde: <https://www.mapbox.com/studio/styles>.

## Añadir un Marcador

Para añadir un Marcador se usa el método `addMarker()`:

```
IconFactory iconFactory = IconFactory.getInstance(MainActivity.this);  
Icon icon = iconFactory.fromResource(R.drawable.taxismall2);  
MarkerOptions markerSanJacinto = new MarkerOptions()  
    .position(new LatLng(-21.585975, -64.700308))  
    .title("San Jacinto")  
    .setIcon(icon);
```

## Añadir movimiento de Cámara

Para añadir un movimiento de Cámara con animación podemos usar:

```
mapboxMap.animateCamera(CameraUpdateFactory.newLatLngZoom(new LatLng(-21.585975,  
-64.700308), 12), 12000);
```



Figure 9: App Mapbox en funcionamiento

Toda las funcionalidades y documentación de Mapbox se encuentra en: <https://docs.mapbox.com/>

From:  
<http://wiki.local/> - **Wiki.Local**

Permanent link:  
[http://wiki.local/doku.php?id=materias:tecnologias-emergentes:unidad\\_1:17\\_mapas\\_en\\_android\\_kotlin](http://wiki.local/doku.php?id=materias:tecnologias-emergentes:unidad_1:17_mapas_en_android_kotlin)

Last update: **2024/04/30 21:01**

