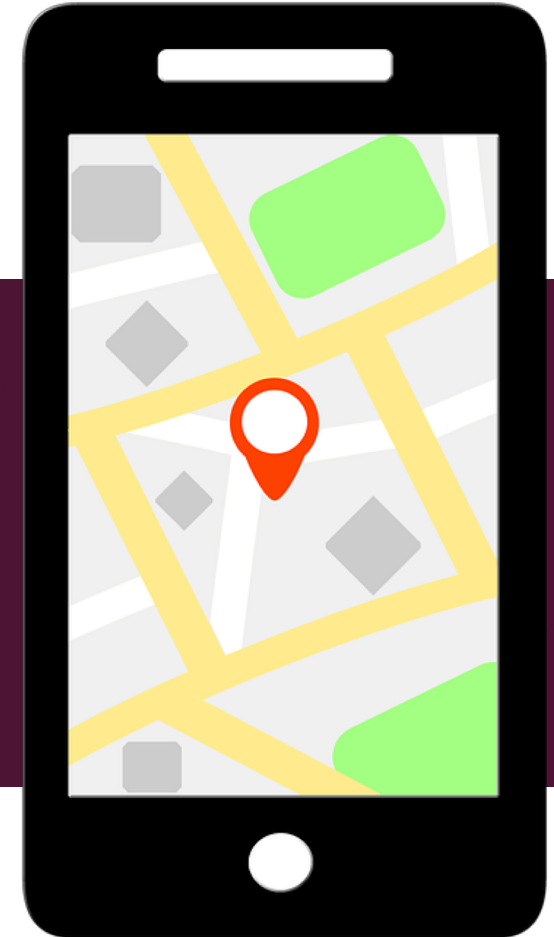


# API DE GEOLOCALIZACIÓN

PROGRAMACIÓN EN ANDROID



# ÍNDICE

**1** Introducción

---

**2** Implementación paso a paso de los casos de uso

---

**3** Demostración de uso

---

**4** Extras implementados

---

**5** Referencias

---

# INTRODUCCIÓN

Una de las características únicas de las aplicaciones móviles es el **reconocimiento de ubicación**. Las personas llevan sus dispositivos móviles con ellos a todas partes, y agregar el reconocimiento de ubicación a su aplicación ofrece a los usuarios una **experiencia más contextual**.

**INSERTAR UN MAPA EN TU APLICACIÓN**

**OBTENER DATOS DE LOCALIZACIÓN PERIÓDICAMENTE**




# Implementación Paso A Paso

---

**PASOS COMUNES PARA AMBOS CASOS**

## OBTENER KEY DE GOOGLE PLAY SERVICES

La clave API se usa para rastrear las solicitudes de API asociadas y así poder controlar su **uso y facturación**.

1. Ir a la consola de **Google Cloud Platform**.
2. En el menú desplegable Proyecto, seleccionar o crear el proyecto para el que deseamos agregar una clave API.
3. En el menú de navegación,  seleccionar **API y servicios > Credenciales**.
4. En la página **Credenciales**, hacer clic en **Crear credenciales > clave API**. El cuadro de diálogo **creado por la clave API** muestra la nueva clave creada.



2

PANEL DE CONTROL

ACTIVIDAD



Información del proyecto

Nombre de proyecto

ApiGeolocalizacion

ID del proyecto

apigeolocalizacion-225115

Número del proyecto

677199970816



Ir a la configuración del proyecto



Recursos

Este proyecto no tiene recursos



Traza

1



Inicio



Mercado



Facturación



APIs y servicios



Asistencia



IAM y administración



Primeros pasos



Seguridad

COMPUTE



App Engine



Compute Engine

Panel de control

Biblioteca

Credenciales

3

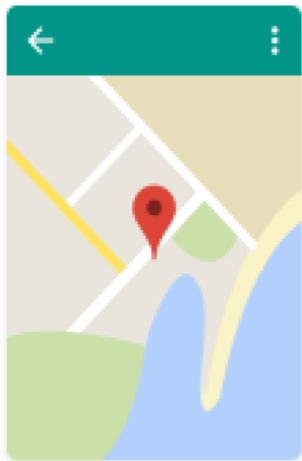


# Implementación Paso A Paso

---

**INSERTAR UN MAPA EN TU APLICACIÓN**

## PASO 1 | CREAR EL PROYECTO



Google Maps Activity

```
google_maps_api.xml x
1  <resources>
2    <string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">YOUR KEY HERE</string>
3  </resources>
4
```

Al crear el proyecto a partir de este tipo de actividad, Android Studio nos genera dentro de la carpeta **values** un fichero llamada **google\_maps\_api.xml**, el cuál se despliega automáticamente al comenzar la edición. En este fichero, debemos pegar la **KEY** que generamos anteriormente.





# Implementación de la Actividad

---

## **PASO 2**

```
1 public class Mapas extends AppCompatActivity implements
2     GoogleMap.OnMyLocationButtonClickListener,
3     GoogleMap.OnMyLocationClickListener,
4     OnMapReadyCallback {
5
6     private GoogleMap mapa;
7     private SupportMapFragment fragmentMapa;
8     private Geocoder geocoder;
9
10    @Override
11    protected void onCreate(Bundle savedInstanceState) {
12        super.onCreate(savedInstanceState);
13        setContentView(R.layout.activity_mapas);
14        setFragmentMapa((SupportMapFragment) getSupportFragmentManager().findFragmentById(R.id.fragment_mapa));
15        setGeocoder(new Geocoder(getApplicationContext()));
16        getFragmentMapa().getMapAsync(this);
17    }
```

```
1  @Override
2  public void onMapReady(GoogleMap googleMap) {
3      setMapa(googleMap);
4      asignarEventos();
5      activarMiLocalizacion();
6  }
7
8
9  private void asignarEventos() {
10     getMapa().setOnMyLocationButtonClickListener(this);
11     getMapa().setOnMyLocationClickListener(this);
12     getEditTextDireccion().setOnKeyListener(this);
13 }
```

<uses-permission android:name="android.permission.ACCESS\_FINE\_LOCATION" />

```
1 private void solicitarPermisosGeolocalizacion() {
2     String[] permisos = {Manifest.permission.ACCESS_FINE_LOCATION};
3     if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
4         if (ContextCompat.checkSelfPermission(this, permisos[0]) != PackageManager.PERMISSION_GRANTED) {
5             requestPermissions(permisos, CODIGO_SOLICITUD_PERMISOS_GEOLOCALIZACION);
6         }
7         else {
8             getMapa().setMyLocationEnabled(true);
9         }
10    }
11    else {
12        getMapa().setMyLocationEnabled(true);
13    }
14 }
```

```
1
2 @Override
3 public boolean onMyLocationButtonClick() {
4     return false;
5 }
6
7 @Override
8 public void onMyLocationClick(@NonNull Location location) {
9     String mensaje = "Localización actual: " + location.getLatitude() + ", " + location.getLongitude();
10    final Bundle argumentos = new Bundle(1);
11    argumentos.putString(MENSAJE_NOTIFICACION, mensaje);
12    DialogFragmentNotificaciones dialogNotificaciones = new DialogFragmentNotificaciones();
13    dialogNotificaciones.setArguments(argumentos);
14    dialogNotificaciones.show(getSupportFragmentManager(), TAG_DIALOG_FRAGMENT_NOTIFICACIONES);
15 }
16
```



# Implementación Paso A Paso

---

**OBTENER DATOS DE LOCALIZACIÓN  
PERIÓDICAMENTE**

## PASO 1 | ELECCIÓN DEL PROVEEDOR DE LOCALIZACIÓN

Las aplicaciones pueden aprovechar las señales proporcionadas por **varios sensores** en el dispositivo para determinar la ubicación del dispositivo. **Sin embargo, elegir la combinación correcta no es sencillo.** Encontrar una solución que también sea **eficiente en batería** es aún más complicado.

### PROVEEDOR DE UBICACIÓN FUSIONADA

El proveedor de ubicación fusionada **administra las tecnologías de ubicación anteriormente descritas**, y proporciona una API simple en la que se puede especificar la calidad de servicio requerida. Por ejemplo, puedes solicitar los datos **más precisos disponibles**, o la **mejor precisión posible** sin consumo de energía adicional.



# Implementación de la Actividad

---

## **PASO 2**



```
1 public class Localizacion extends AppCompatActivity implements OnSuccessListener<Location>,
2                                     GoogleApiClient.OnConnectionFailedListener,
3                                     GoogleApiClient.ConnectionCallbacks {
4
5     private FusedLocationProviderClient proveedorLocalizacion;
6     private LocationRequest solicitudLocalizacion;
7     private LocationCallback callbackLocalizacion;
8     private Location localizacionActual;
9
10    @Override
11    protected void onCreate(Bundle savedInstanceState) {
12        super.onCreate(savedInstanceState);
13        setContentView(R.layout.activity_localizacion);
14        setProveedorLocalizacion(LocationServices.getFusedLocationProviderClient(this));
15        solicitarPermisosGeolocalizacion();
16    }
17 }
```

```
1 private void solicitarPermisosGeolocalizacion() {
2     String[] permisos = {Manifest.permission.ACCESS_FINE_LOCATION};
3     if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {
4         if (ContextCompat.checkSelfPermission(this, permisos[0]) != PackageManager.PERMISSION_GRANTED) {
5             requestPermissions(permisos, Mapas.CODIGO_SOLICITUD_PERMISOS_GEOLOCALIZACION);
6         }
7         else {
8             getProveedorLocalizacion().getLastLocation().addOnSuccessListener(this);
9         }
10    }
11    else {
12        getProveedorLocalizacion().getLastLocation().addOnSuccessListener(this);
13    }
14 }
```

```
1 @Override
2 public void onClick(View vista) {
3     if (vista.getId() == R.id.boton_iniciar_recepcion) {
4         crearSolicitudLocalizacion();
5         inicializarCallbackLocalizacion();
6         iniciarRecepcionLocalizacion();
7     }
8 }
```

```
1 protected void crearSolicitudLocalizacion() {  
2     setSolicitudLocalizacion(LocationRequest.create());  
3     getSolicitudLocalizacion().setInterval(INTERVAL_LOCALIZACION);  
4     getSolicitudLocalizacion().setFastestInterval(FASTEST_INTERVAL_LOCALIZACION);  
5     getSolicitudLocalizacion().setPriority(LocationRequest.PRIORITY_HIGH_ACCURACY);  
6 }
```

## PRIORITY

Este método establece la **prioridad de la solicitud**, lo que le da a los servicios de ubicación de los servicios de Google Play una **sugerencia sólida sobre qué fuentes de ubicación usar**. Los siguientes valores son compatibles:

PRIORITY\_HIGH\_ACCURACY | PRIORITY\_LOW\_POWER | PRIORITY\_NO\_POWER  
PRIORITY\_BALANCED\_POWER\_ACCURACY

```
1 private void inicializarCallbackLocalizacion() {
2     setCallbackLocalizacion(new LocationCallback() {
3         @Override
4         public void onLocationResult(LocationResult locationResult) {
5             if (locationResult != null) {
6                 for (Location location: locationResult.getLocations()) {
7                     try {
8                         setLocalizacionActual(location);
9                         mostrarDatosLocalizacion();
10                    } catch (IOException e) {
11                        e.printStackTrace();
12                    }
13                }
14            }
15        }
16    });
17 }
```

1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16  
17

```
private void iniciarRecepcionLocalizacion() {  
    if (Build.VERSION.SDK_INT >= Build.VERSION_CODES.M) {  
        if (ContextCompat.checkSelfPermission(Localizacion.this, Manifest.permission.ACCESS_FINE_LOCATION) == PackageManager.PERMISSION_GRANTED) {  
            getProveedorLocalizacion().requestLocationUpdates(getSolicitudLocalizacion(), getCallbackLocalizacion(), null);  
            Toast.makeText(this, "Recepción Iniciada", Toast.LENGTH_SHORT).show();  
        }  
    }  
}
```



# Uso del Geocoder y Marker

---

**EXTRAS IMPLEMENTADOS**

```
1  private LatLng obtenerCoordenadas(String direccion) throws IOException {
2      List<Address> listLocalizaciones = getGeocoder().getFromLocationName(direccion, 1);
3      Address localizacion = listLocalizaciones.get(0);
4      double latitud = localizacion.getLatitude();
5      double longitud = localizacion.getLongitude();
6      return new LatLng(latitud, longitud);
7  }
8
9  private void buscarDireccion(String direccion) throws IOException {
10     LatLng coordenadas = obtenerCoordenadas(direccion);
11     if (coordenadas != null) {
12         setDireccionBuscada(coordenadas);
13         CameraUpdate cameraUpdate = CameraUpdateFactory.newLatLngZoom(coordenadas, ZOOM);
14         getMapa().moveCamera(cameraUpdate);
15     }
16 }
```



1  
2  
3  
4  
5  
6  
7  
8  
9  
10  
11  
12  
13  
14  
15  
16

```
private String obtenerDireccion() throws IOException {  
    List<Address> listLocalizaciones = getGeocoder().getFromLocation(latitudActual, longitudActual, 1);  
    String direccion = null;  
    if (listLocalizaciones.size() > 0) {  
        Address localizacion = listLocalizaciones.get(0);  
        direccion = localizacion.getAddressLine(0);  
    }  
    else {  
        Toast.makeText(this, "Dirección no encontrada", Toast.LENGTH_SHORT).show();  
    }  
    return direccion;  
}
```

```
1  @Override
2  public boolean onKey(View view, int keyCode, KeyEvent event) {
3      if (view.getId() == getEditTextDireccion().getId()) {
4          if ((event.getAction() == KeyEvent.ACTION_DOWN) && (keyCode == KeyEvent.KEYCODE_ENTER)) {
5              String direccion = getEditTextDireccion().getText().toString();
6              if (!direccion.equals(DIRECCION_VACIA)) {
7                  try {
8                      buscarDireccion(direccion);
9                  } catch (IOException e) {
10                     e.printStackTrace();
11                 }
12             }
13             return true;
14         }
15     }
16     return false;
17 }
```

## REFERENCIAS

1. [ANDROID DEVELOPERS | SOLICITUD DE LOCALIZACIÓN](#)
2. [ANDROID DEVELOPERS | OBTENER ÚLTIMA LOCALIZACIÓN CONOCIDA](#)
3. [ANDROID DEVELOPERS | OBTENER LOCALIZACIÓN PERIÓDICAMENTE](#)
4. [ANDROID DEVELOPERS | INSERTAR UN MAPA](#)
5. [FLIPANDROID | TIPOS DE PROVEEDORES DE LOCALIZACIÓN](#)