

Software para Dispositivos Móviles
Grado en Ingeniería Informática del Software
Escuela de Ingeniería Informática – Universidad de Oviedo

Geolocalización en Android

Juan Ramón Pérez Pérez
Departamento de Informática
jrpp@uniovi.es

Experiencia basada en el contexto

- Una de las características únicas de las aplicaciones móviles es el **conocimiento de la ubicación**.
- Los usuarios llevan su **dispositivo móvil a todos sitios**
- Para ofrecer una experiencia contextual podemos añadir la **conciencia de la ubicación** en las aplicaciones que desarrollemos.

La capa "My Location"

```
// Activa la capa de geolocalización
```

```
mapa.setMyLocationEnabled(true);
```

- Es necesario disponer de permisos
- No permite recuperar datos por programa
- Sólo capa visual con el punto indicando dónde nos encontramos



Alternativas uso geolocalización en Android

- **Android framework location APIs**, package android.location
- **Google Play Services Location API**, parte de Google Play Services
- Google aconseja el uso de la segunda alternativa:
 - Ofrece un framework de más alto nivel y más potente.
 - Automatiza tareas de elección de proveedor de localización y gestión de energía
 - Incorpora nuevas características como detección de la actividad



Google Play Services Location API

Enlazar Google Play Services (Android Studio)

- Forma parte de Google Play Services
- Mismo Proceso previo que la API Google Maps
 1. build.gradle (Module: app)
 2.

```
Implementation ("com.google.android.gms:play-services-location:<versión>.0.0")
```
 3. Sync Project with Gradle Files



Permisos para Servicios de localización

- Permisos que necesitamos incorporar en el AndroidManifest.xml

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

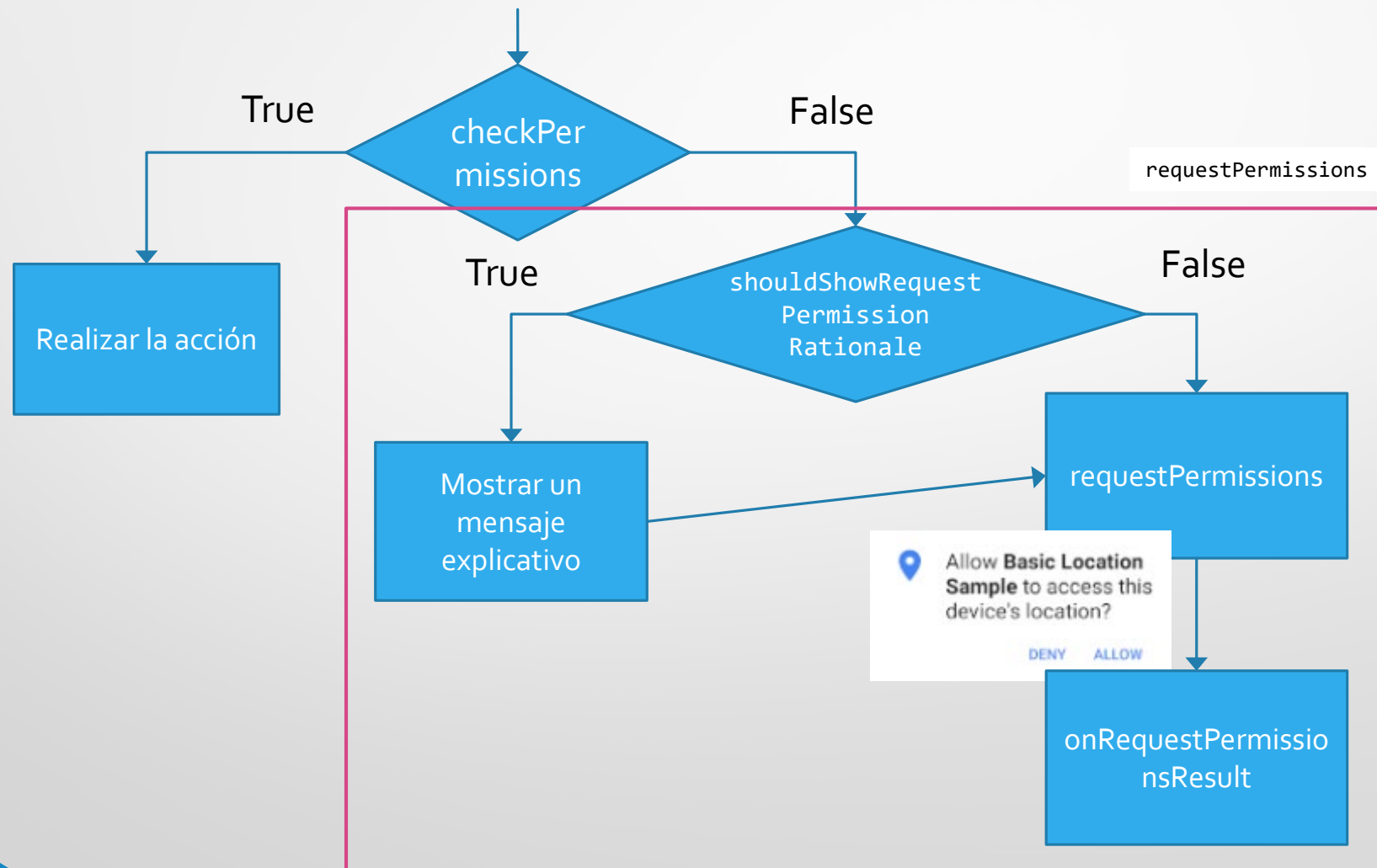
```
<uses-permission android:name="android.permission.ACCESS_FINE_LOCATION" />
```

- Cada permiso está relacionado con un *location provider*:
 - **ACCESS_FINE_LOCATION** : Localización más exacta (Sistema de posicionamiento global)
 - **ACCESS_COARSE_LOCATION** : Localización menos exacta (Ej. Red, triangulación de la posición,..)

Gestionar permisos (I)

- API 23 (6.0) o superior → pedir permisos en tiempo de ejecución
 - Mayor control sobre la funcionalidad: elegir y revocar permisos
- Permisos normales y críticos (peligrosos)
- La gestión de permisos se deja bajo la responsabilidad del desarrollador
- Hay que intentar adaptar la aplicación a los permisos concedidos
- Un permiso puede ser denegado en cualquier momento

Gestionar permisos (II) – Diagrama de estados



Cómo obtener la ubicación actual

- **Location services guarda** automáticamente la **ubicación** actual del usuario.
- Las aplicaciones pueden solicitarla cuando la necesiten
- Las APIs de localización de Google Play Services utilizan un proveedor de localización **"combinado"** que utiliza información de distintas fuentes: wifi, GPS, etc.
- El proveedor de localización funciona como un **servicio** y las aplicaciones como **clientes**

Creación de una instancia de GoogleApi (Kotlin)

- Google Play Services tiene una forma común de crear clientes para utilizar las distintas APIs
- Es a través de subclases: **GoogleApi**
- En nuestro caso: **FusedLocationProviderClient**

```
val cliente: FusedLocationProviderClient =  
    LocationServices.getFusedLocationProviderClient(context)
```

GoogleApi

- Las subclases gestionan la conexión entre la App y los servicios de Google Play
- Se encargan de encolar las llamadas hasta que se establezca una conexión real
- Las instancias son:
 - “Baratas” de crear
 - Thread-safe
 - Unifica duplicados automáticamente
 - Reconecta cuando sea necesario

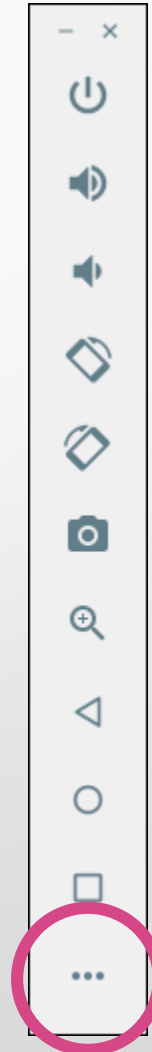
Obtener la última ubicación conocida (Kotlin)

```
cliente.lastLocation
    .addOnSuccessListener { location: Location? ->
        if (location != null) {
            // Logica manejar localización
        }
    }
```

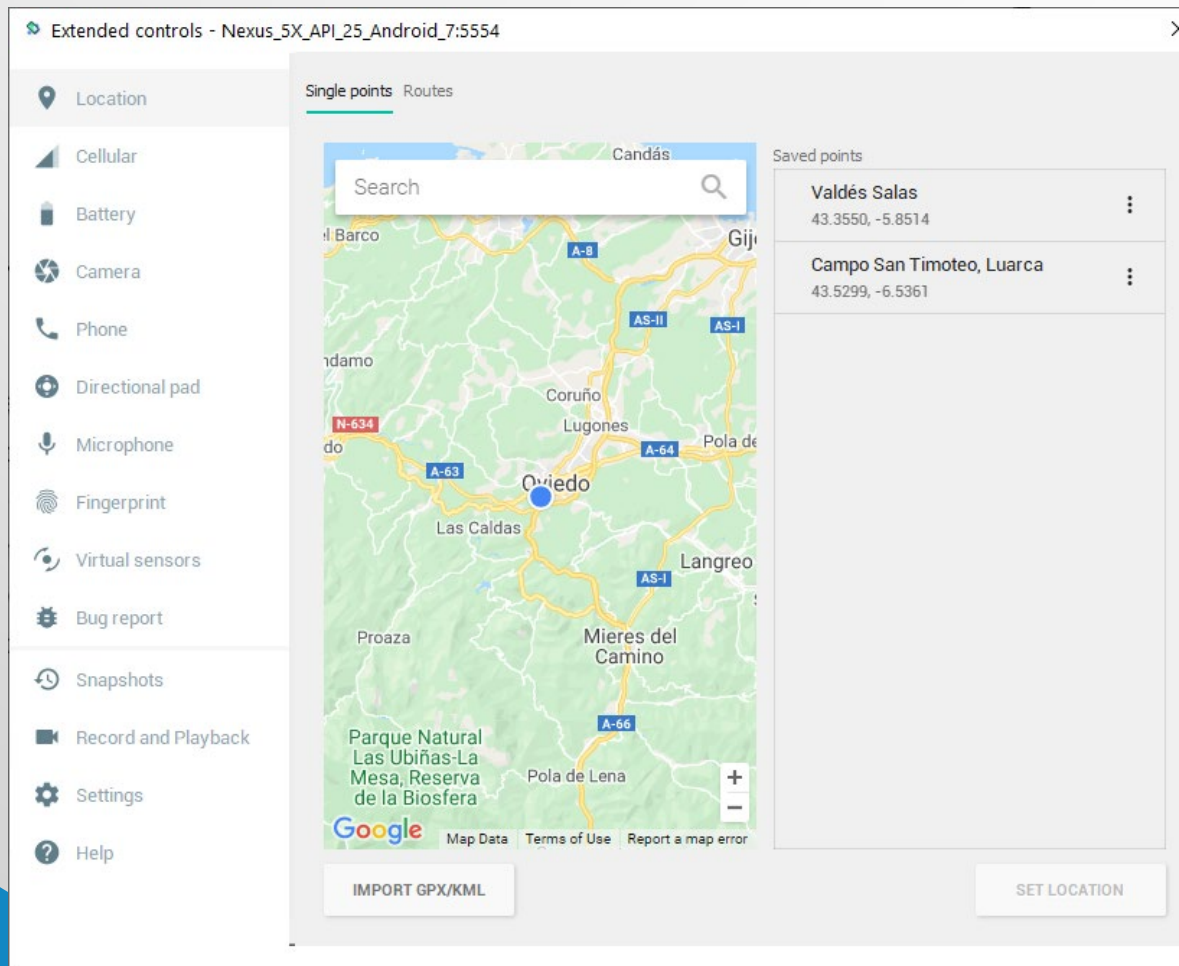
- Devuelve resultado de tipo Location
- De un objeto [Location](#), entre otras cosas, podemos saber:
 - Posición : Latitud y Longitud
 - Fecha de la última localización

Datos de localización simulados con el emulador (Android Studio)

- Controles del emulador > Opción más controles
- Permite controlar diversas condiciones del dispositivo
 - Llamadas, batería, señal y sensores
- Location, permite controlar valores del GPS:
 - como coordenadas individuales latitud / longitud,
- Como secuencia de coordenadas
 - con un fichero GPX para reproducir la ruta o
 - un fichero KML para introducir múltiples puntos




Configuración de la localización en el emulador



- ¿Qué podemos simular?
- Sólo podemos fijar el que simula GPS (FINE_LOCATION)
- Podemos simular rutas donde la posición va cambiando en el tiempo

Pruebas en dispositivos reales simulando ubicaciones

- Los dispositivos Android se pueden configurar para sus fuentes de localización proporcionen datos simulados
- Una vez configurado el dispositivo podemos desarrollar aplicaciones que generen estos datos
- Un ejemplo de aplicación es:
 - Fake GPS location



Obtener dirección:
Servicio Geocodificación mediante tarea
asíncrona

Servicios de Google Maps: Geocoder

- Dos funciones:
 - Convertir direcciones a coordenadas geográficas (Geocodificación)
 - Convertir coordenadas geográficas a direcciones postales (Geocodificación inversa)


<https://maps.googleapis.com/maps/api/geocode/json?latlng=43.354971,-5.851552>
- Android encapsula la comunicación con el servicio en la **clase Geocoder** (android.location.Geocoder)

Clase Geocoder

- Geocoding: dirección → coordenadas geográficas
 - `List<Address> getFromLocationName(String locationName, int maxResults, double lowerLeftLatitude, double lowerLeftLongitude, double upperRightLatitude, double upperRightLongitude)`
 - `List<Address> getFromLocationName(String locationName, int maxResults)`
- Geocoding inverso: coordenadas → dirección
 - `List<Address> getFromLocation(double latitude, double longitude, int maxResults)`

Llamada al servicio geocoding

- **getFromLocation()**
- Es una llamada **síncrona**, hace la petición y queda bloqueado hasta que recibe respuesta
- Podría **no ser inmediata** dejando bloqueada la interfaz.
- **No se debe llamar desde el mismo hilo de la interfaz de usuario** de la aplicación.



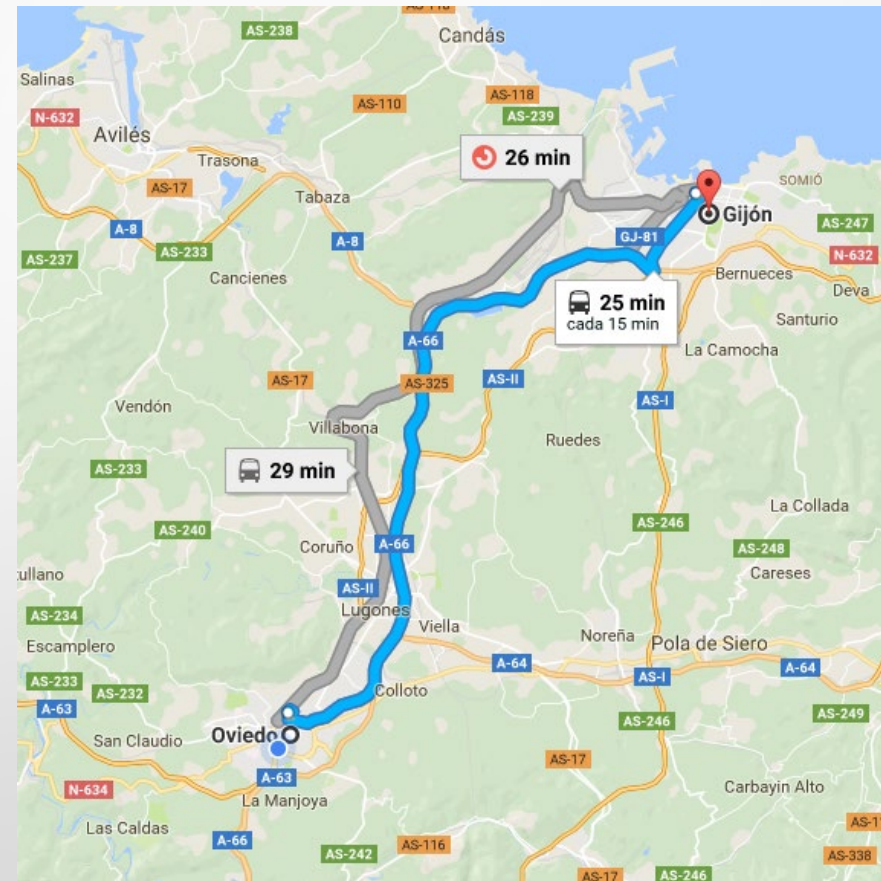
Otros servicios de información geográfica

Otros servicios de información geográfica

- Distintos servicios REST relacionados con información geográfica
- Devuelven la respuesta en XML o JSON
- Google no proporciona un framework para encapsularlos
- Llamar al servicio REST → Procesar la respuesta JSON

Servicios *Directions*

- Servicio que busca la mejor ruta entre dos puntos
- Gran versatilidad: caminando – en coche, puntos intermedios, rutas alternativas
- [Oviedo – Gijón](#)
- <https://maps.googleapis.com/maps/api/directions/json?origin=Oviedo+Asturias&destination=Gijon+Asturias>



Otros servicios

- Elevation
- Distance Matrix
- Roads
- Time Zone
- Places

<https://developers.google.com/maps/documentation/webservices/?hl=es>



Detectar proximidad a un punto

Geofencing:

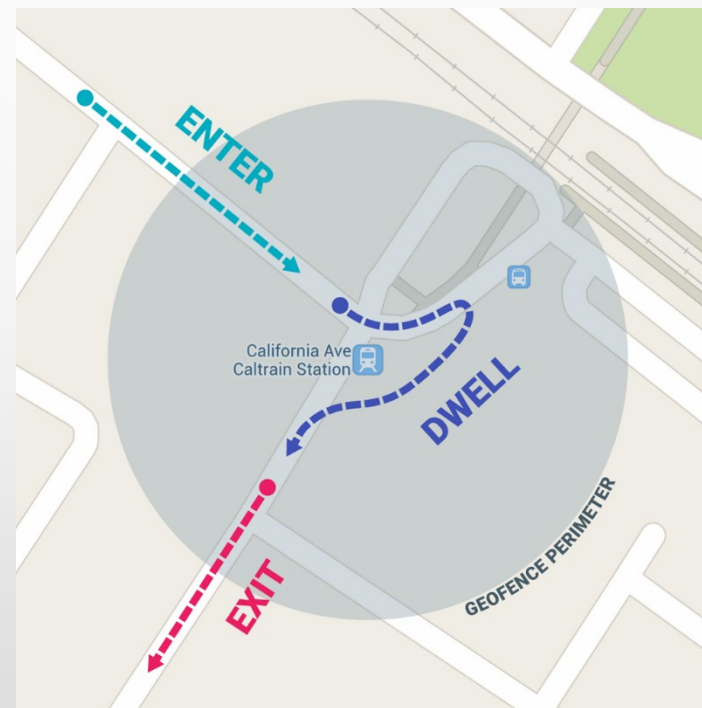
Detectar proximidad a un punto

- Utiliza la conciencia de la localización actual del usuario combinada con la proximidad a un punto fijo.
- Área circular virtual establecida alrededor de un punto geográfico (*geofence*)
 - Se define por un punto central (Lat, Lng) y un radio
- Google Services permite definir y monitorizar estos perímetros

Características *geofences*

- Se pueden activar **múltiples** *geofences*
- Se pueden recoger **eventos** al **entrar y salir** en un geofence
- O establecer un **tiempo en el interior** de un geofence
- Además, se puede establecer un tiempo de duración, a partir del cual se eliminan automáticamente

<http://developer.android.com/intl/es/training/location/geofencing.html>



Referencias

- Guías sobre Google Maps API para Android:
 - <https://developers.google.com/maps/documentation/android/map>
- Tutorial Android Developer: Making Your App Location-Aware
 - <http://developer.android.com/training/location/index.html>