**GOOGLE MAPS API v2**

Google Maps nos proporciona un servicio de cartografía *online* que podremos utilizar en nuestras aplicaciones Android.

**Principales novedades respecto a anteriores versiones:**

- Integración con los Servicios de Google Play (Google Play Services) y la Consola de APIs.

- Utilización a través de un nuevo tipo específico de fragment *(*MapFragment*).*

- Utilización de *mapas vectoriales*, lo que repercute en una mayor velocidad de carga y una mayor eficiencia en cuanto a uso de ancho de banda.

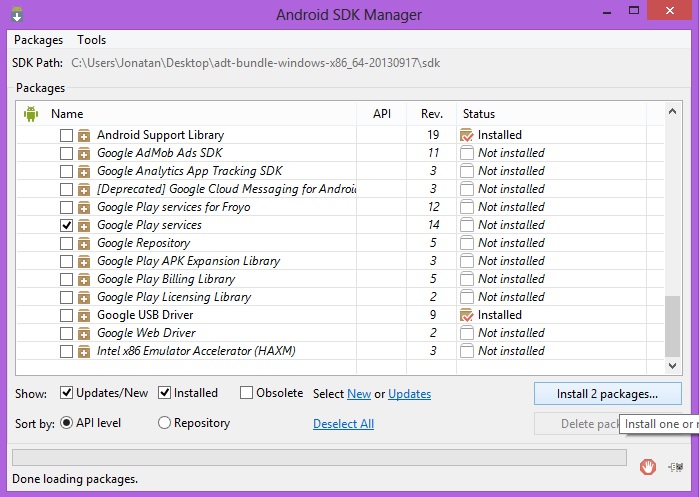
- Mejoras en el sistema de caché, lo que reducirá en gran medida las áreas en blanco que tardan en cargar.

- Los mapas son ahora 3D, es decir, podremos mover nuestro punto de vista de forma que lo veamos en perspectiva.

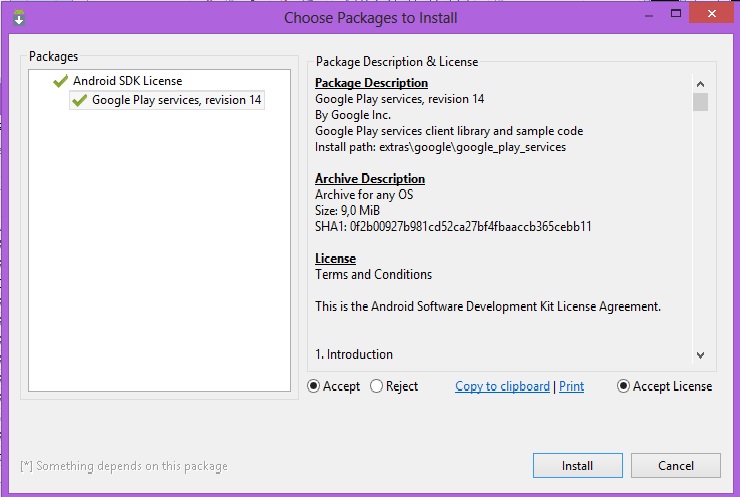
 - Tiene un inconveniente que esta versión solo funciona en el dispositivo con Google Play instalado.

**LIBRERÍA GOOGLE-PLAY-SERVICES**

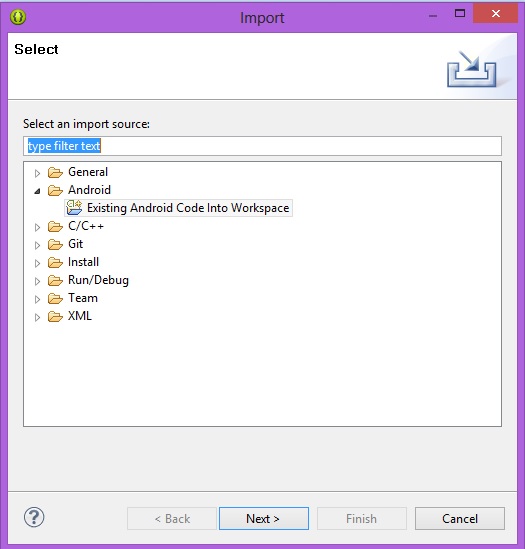
Google-Play-Services 🡪 En esta versión de la API de Google Maps es totalmente obligatorio tener las librerías google-play-services\_lib importadas en el proyecto.

Por lo tanto primero tenemos que descargar el Google Play Services en el SDK. 

Aceptamos las condiciones y le damos a Install.



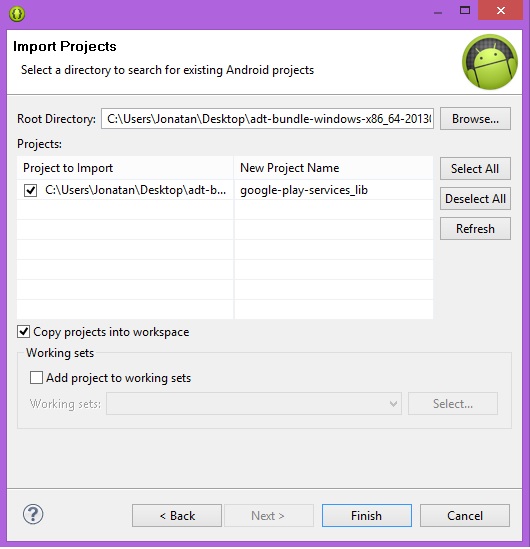
Hecho esto nos vamos a eclipse y en File- Import…

Y vamos a importar los servicios de Google que nos hemos descargado para ello le decimos que es un proyecto de Android.

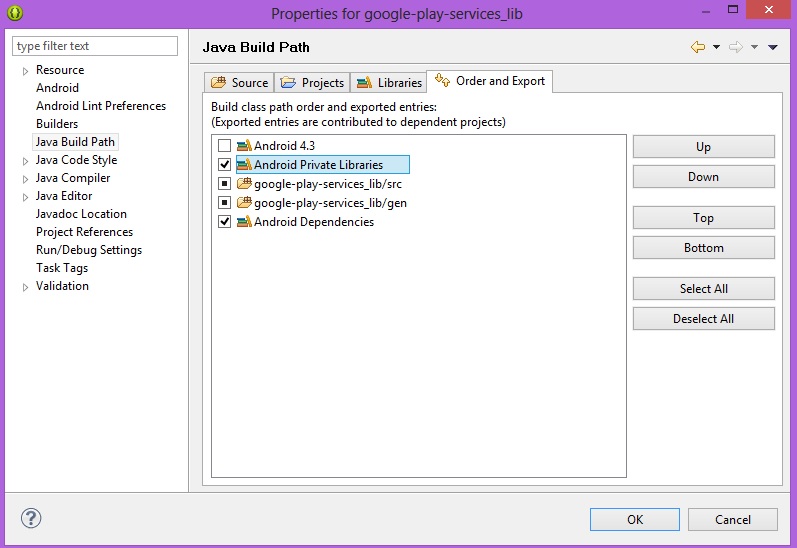
Y seleccionamos el proyecto que se encuentra en la ruta:

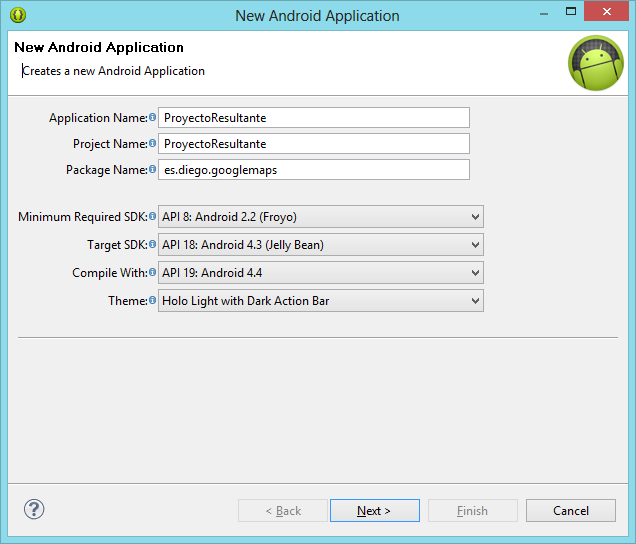
sdk\ extras\google\google\_play\_services\ libproject\google-play-services\_lib

Y copiamos en el Workspace.

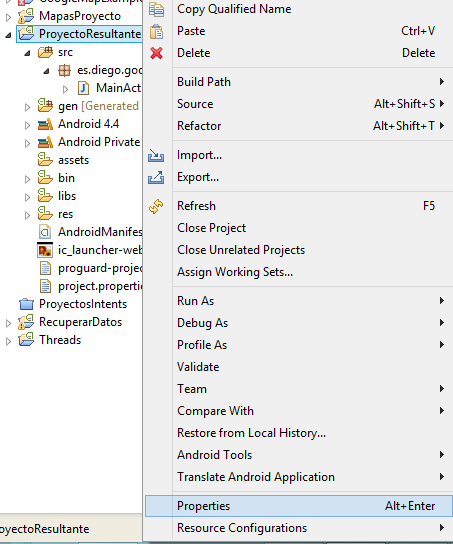


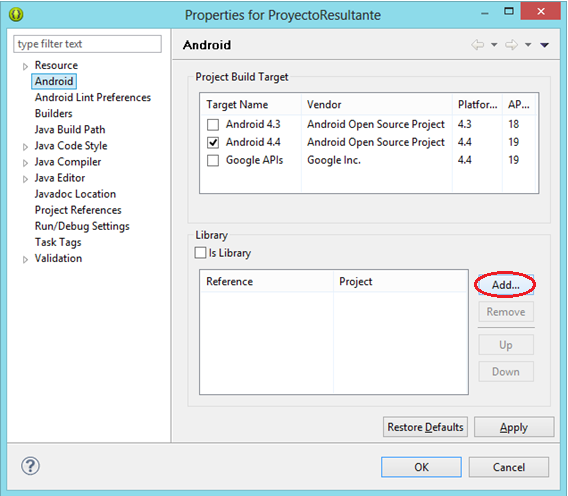
Tenemos que comprobar que el proyecto Google-Play-Services tiene marcadas las opciones Android Private Libraries y Android Dependencies para ello nos iremos al las propiedades del proyecto con el botón derecho Properties y en Java Build Path lo comprobamos.



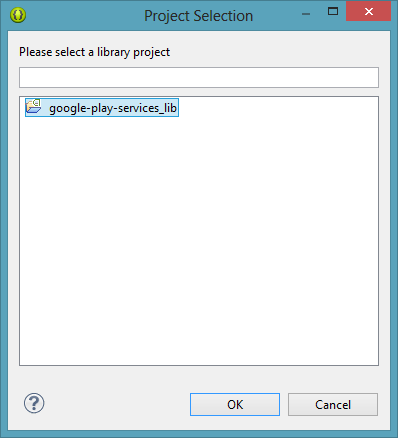
Una vez tengamos esto creamos el proyecto.

Le damos al botón derecho en el proyecto y a Properties

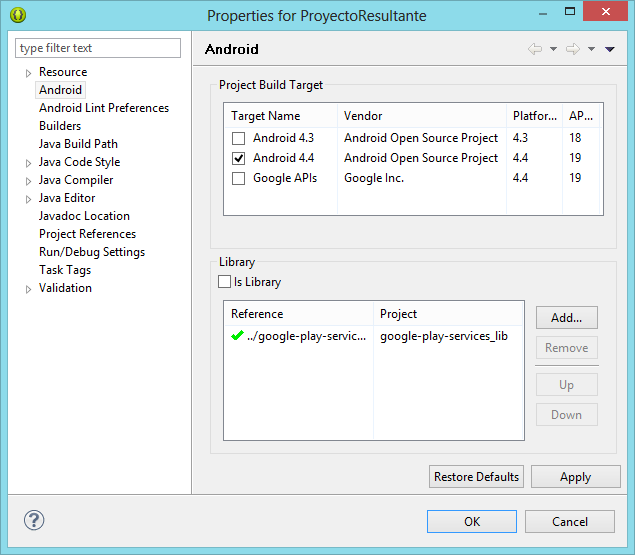


Y en Android le damos a Add…

Nos saldrá esto, seleccionamos el proyecto y le damos a OK.

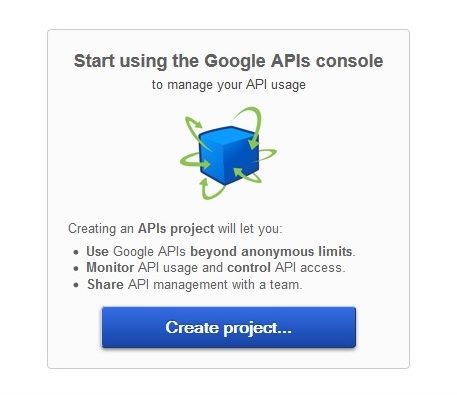


Ya tenemos la librería importada al proyecto.

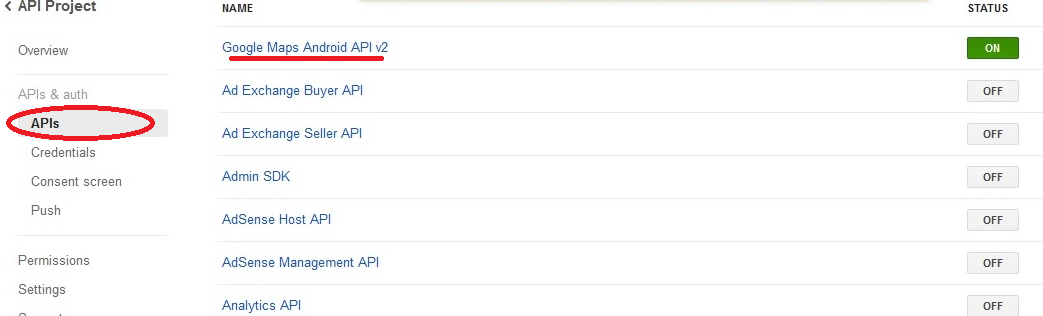


**CREACION GOOGLE API KEY**

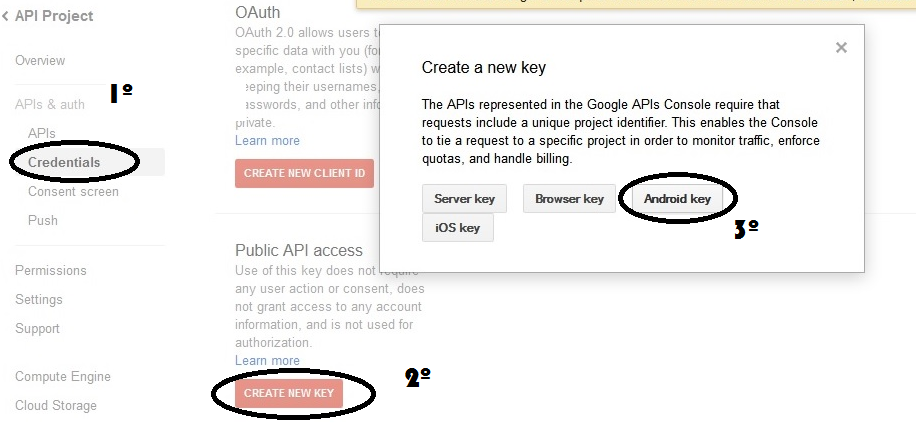
**Para poder utilizar una API de Google es necesario obtener una API-KEY relacionada a nuestro proyecto.**

La primera parte ya estaría, ahora tendríamos que crear la API Key para ello nos vamos a code.google.com/apis/console y creamos un nuevo proyecto con el nombre que queramos.De

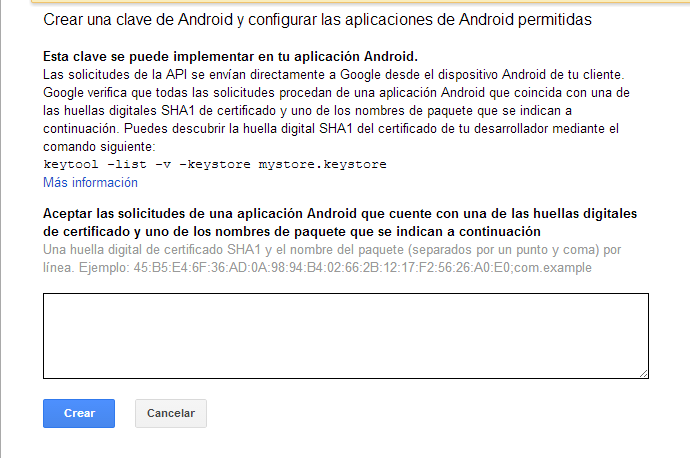
Después nos vamos a APIs&auth y APIs para permitir la utilización de Google Maps Android API v2.



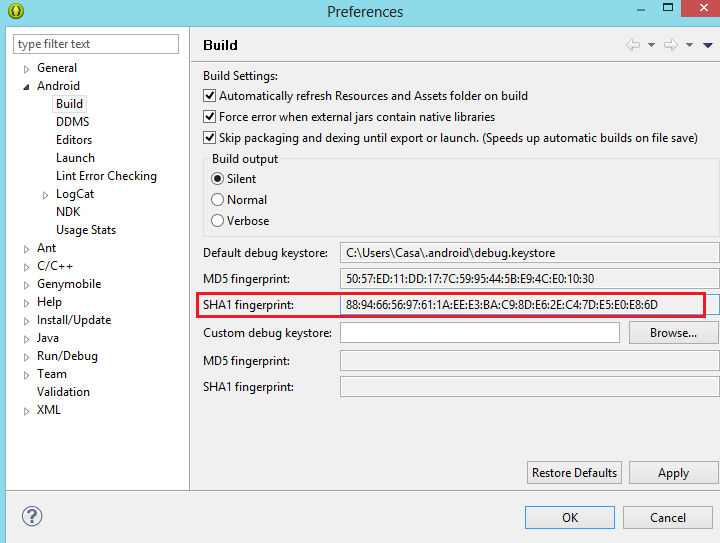
Ahora vamos a crear la API Key para ello nos vamos a Credentials y le damos a CREATE NEW KEY después le especificamos que es una clave para android.



Nos aparecerá esta ventana en ella tenemos que introducir la huella SHA1 seguido de un punto y coma y el nombre del paquete

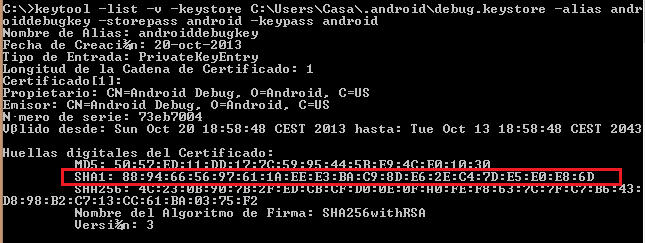


La huella SHA-1 podemos visualizarla de 2 maneras la primera es yendo a eclipse en el menú superior- Window y Preferences , dentro de esta ventana Android- Build y ahí lo vemos, en algunas versiones de eclipse solo podemos visualizar el Default debug keystore que es donde se almacenan estas huellas.

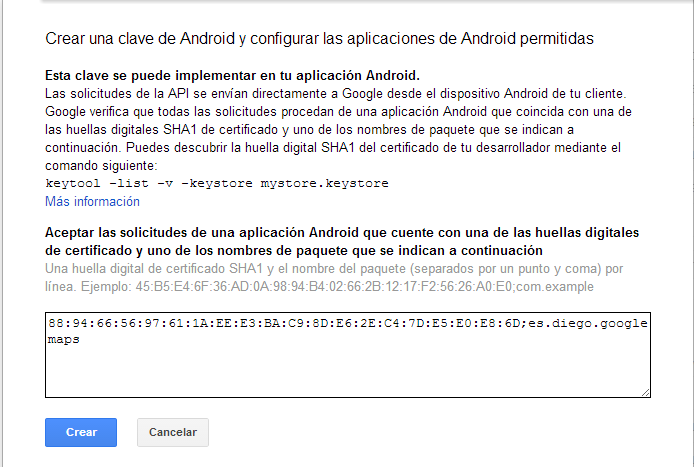


Para visualizar las huellas de ese fichero tenemos que utilizar el programa keytool que está en la carpeta JDK-bin y pondríamos lo siguiente:

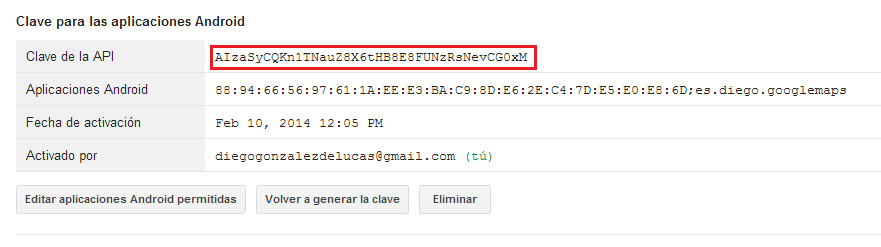
Keytool -list –v –keystore “ruta” – alias androiddebugkey –storepass android –keypass android



Hecho esto ya tenemos lo necesario la Huella y el nombre del paquete que tenemos asi que lo introducimos.



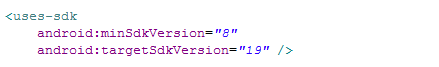
Le damos a crear y observaremos esto. La clave asociada a ese paquete y esa huella esta clave la vamos a tener que usar mas adelante.



DESARROLLO DEL PROYECTO

1º MANIFEST

Primero colocamos la versión SDK minima y la versión tope con la cual nuestra aplicación va a funcionar.



Después colocamos los permisos necesarios que son estos:

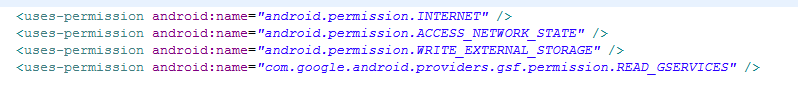
**Internet**🡪 Usado por la API para descargar los mapas de los servidores.

**Access\_Network\_State**🡪 Permite a la API comprobar el estado de la conexión para ver si puede descargar datos.

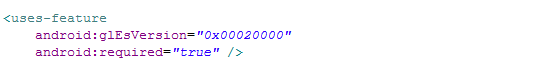
**Write\_External\_Storage**🡪 modificar/eliminar contenido del almacenamiento USB

**Read\_GServices**🡪 Permite a la API acceder a los servicios de Google basados en la web.

Este trozo de código lo colocamos antes de application



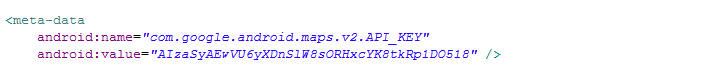
Para la API Maps v2 es necesario tener OpenGL 2 le colocamos este requisito en el Manifest justo después de los permisos.



Dentro de la etiqueta application colocamos este metadata que contiene la versión de la librería de google play services que hemos añadido previamente al proyecto.

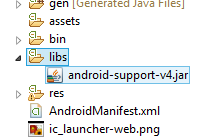


Después colocamos la clave API que hemos obtenido anteriormente de esta manera.



2º LAYOUT

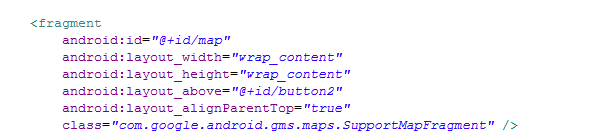
Es importante que tengamos esta librería en la carpeta libs del proyecto ya que incluir la librería android-support nos permite poder utilizarlos en versiones de Android anteriores a la 3.0.



Las versiones más recientes de ADT incluyen por defecto esta librería en nuestros proyectos, pero si no está incluida podéis hacerlo mediante la opción del menú contextual “Android Tools / Add Support Library…”  sobre el proyecto, o bien de forma manual.

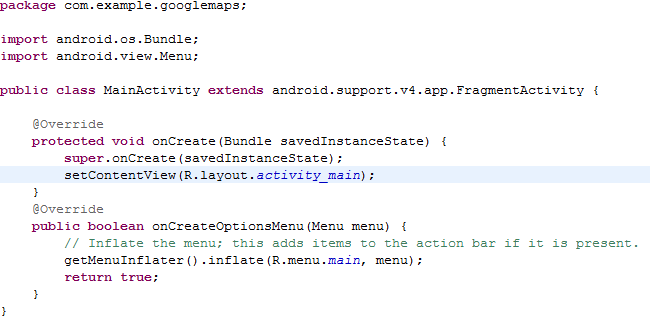
Y el layout es un fragment pero para que sea compatible con todas las versiones le especificamos que

class = “*com.google.android.gms.maps.SupportMapFragment”*

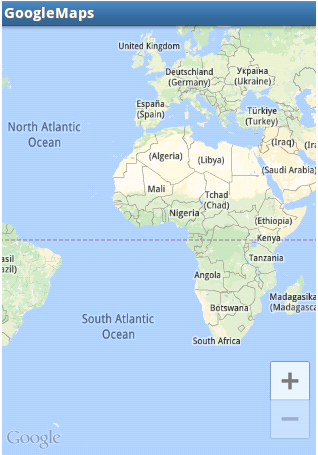


3º MAIN-ACTIVITY

La MainActivity desciende de la clase FragmentActivity que esta en la librería privada **android.support.v4.jar.** Con eso ya valdría.



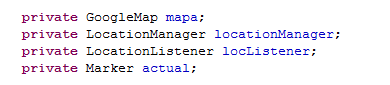
Quedaría así:



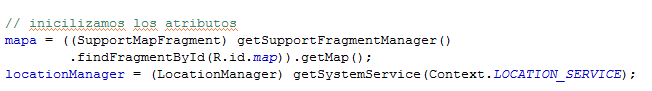
**GPS**

Para usar el gps lo primero es modificar el manifest del proyecto para tener Permisos relativos a la ubicación 

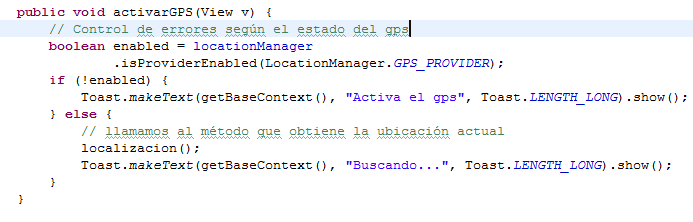
Usaremos la clase **Google Map ,LocationManager y LocationListener** se encargan de administrar lo referente con al gpsy un marker para actualizarlo cada vez que la posición cambie.



Para añadir la funcionalidad del GPS a un mapa tendremos que tener un Objeto **GoogleMap** el cual tomamos del fragment anteriormente visto.



Cuando pulsemos el botón activar que tenemos la activity nos lanza este método.

Activar controlamos los errores que pueden darse si al pulsar el botón no esta activado el gps:

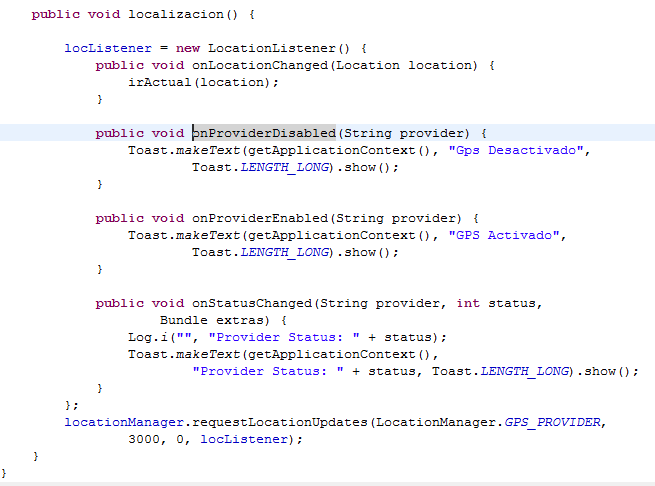
Y si esta activado pues lanza el método localización.De forma que cuando detecte la posición nos va a llevar al método irActual y le pasamos la Location actual. Esto funcionara de forma que cada vez que actualice la posicion nos va a ubicar en el mapa.

Al final de este codigo determinamos las peticiones de actualizar la localizacion entre las cuales pasamos un provider , el tiempo minimo, la distancia minima y el escuchador necesarios para actualizar.

Provider hay dos tipos:

LocationManager.NETWORK\_PROVIDER. Localización por la red de telefonía.

LocationManager.GPS\_PROVIDER. Localización por GPS.



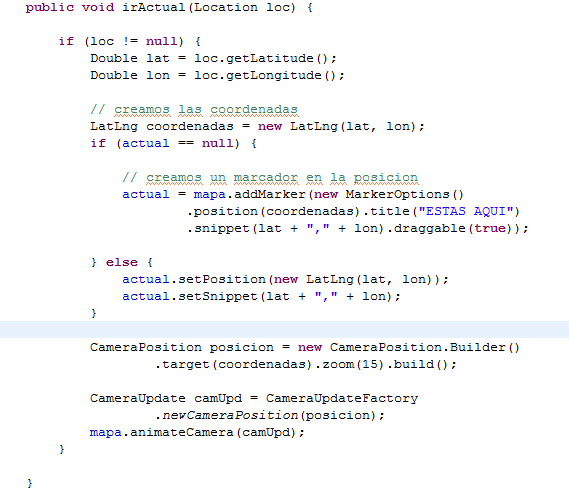
De forma que si el Objeto **Location** no es nulo, tomara los datos de latitud y longitud para crear un objeto **LatLng,** que son coordenadas. La variable actual es un Marker de forma que si es nulo lo creamos añadiendo al mapa un marker al que le pasamos la posicion, le ponemos un titulo y un subtitulo(snippet),en el cual ponemos la latitud y longitud.

Y si no es nulo pues simplemente modifica la posicion del marker y el subtitulo.

Tambien le damos animacion a la camara de forma que cada vez que cambie las coordenadas cambiaremos la posicion de la camara con un objeto **CameraPosition** y pasamos el target que es la posicion y un zoom para que la camara se acerque

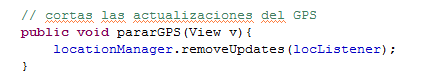
**CameraUpdate** es el objeto que le tenemos que pasar al mapa asi que fabricamos un CameraUpdateFactory.newCameraPosition y le pasamos el CameraPosition

Por ultimo en el mapa le pasamos este objeto CameraUpdate



Este seria el resultado final le pulsaríamos a Activar y va actualizando la posición hasta que le demos a Desactivar.

Desactivar que lleva a un método que elimina las actualización de un escuchador.





BIBLIOGRAFIA

MAPS:

<http://www.sgoliver.net/blog/?p=3244>

<http://www.sgoliver.net/blog/?p=3271>

<http://www.sgoliver.net/blog/?p=3286>

<https://developers.google.com/maps/documentation/android/>

GPS:

<http://www.sgoliver.net/blog/?p=1887>