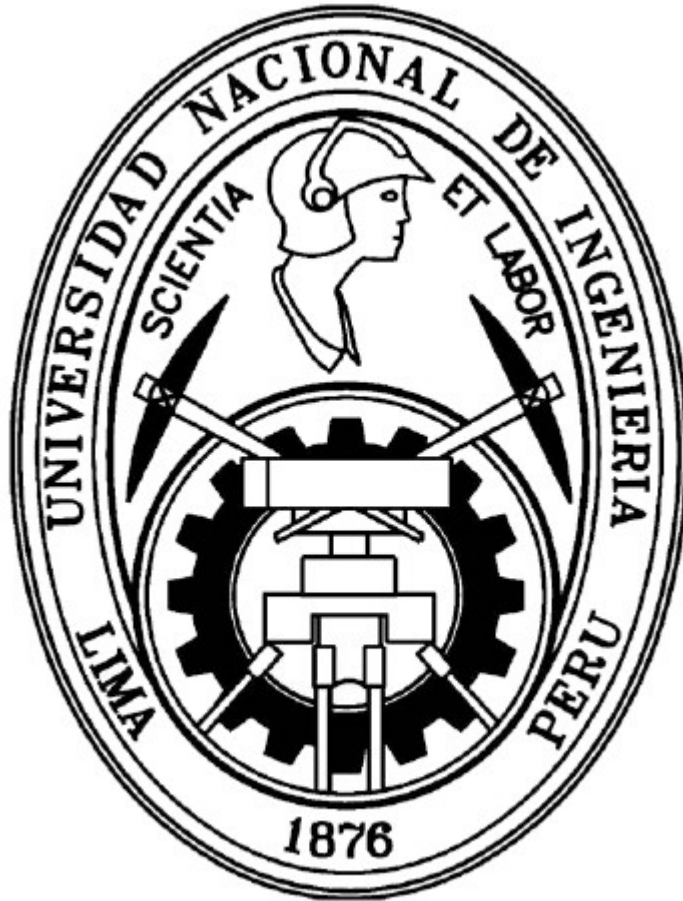


Laboratorio 10.2



Apellidos: Moreno Vera

Nombres: Felipe Adrian

Código: 20120354I

**Asignatura: Programación en Dispositivos Móviles
(CC481)**

2016 - I

Indice

Actividad 1	(3)
--------------------------	------------

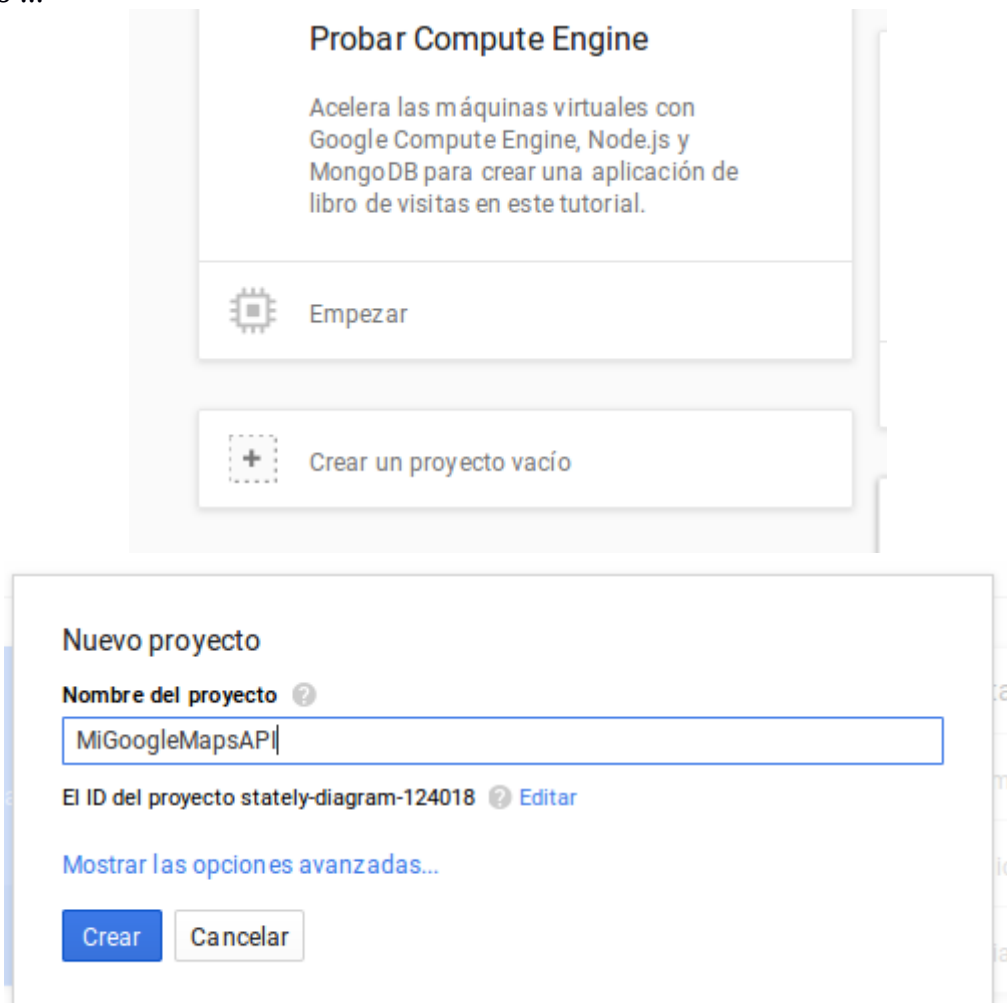
Actividad 1

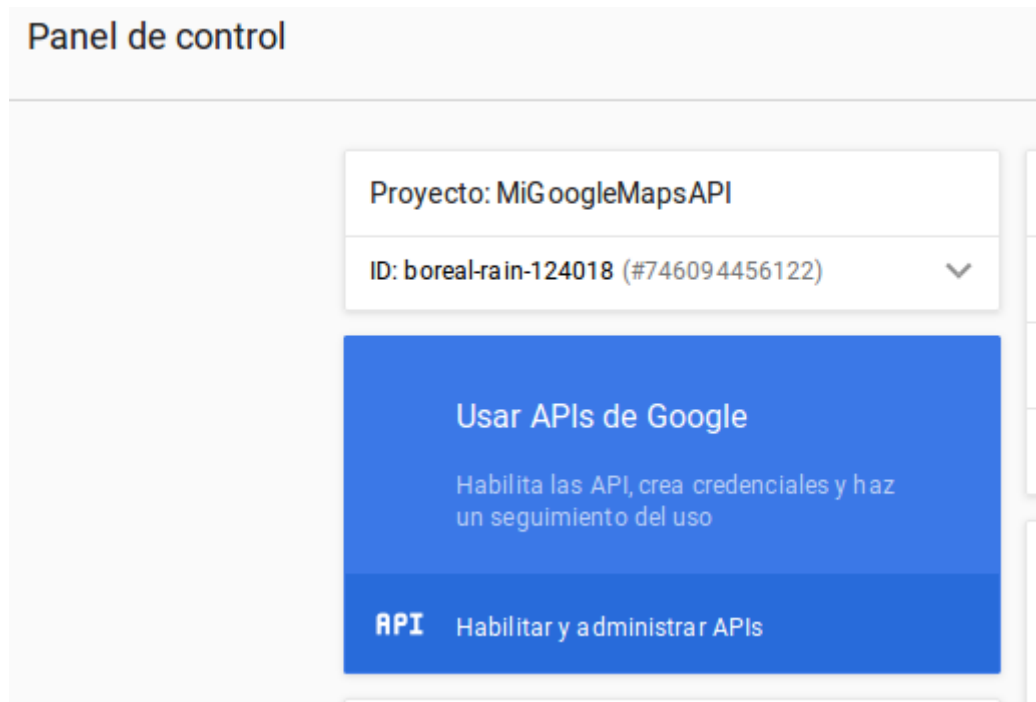
1. Creamos un nuevo proyecto blanco.

2. Como se vio en teoría, lo primero es instalar del SDK Manager “Google Play Services”.

1. El siguiente paso es crear nuestro proyecto en <https://console.developers.google.com/start>

Creando ...





2. Posteriormente buscamos APIs & auth y activamos los siguientes servicios:

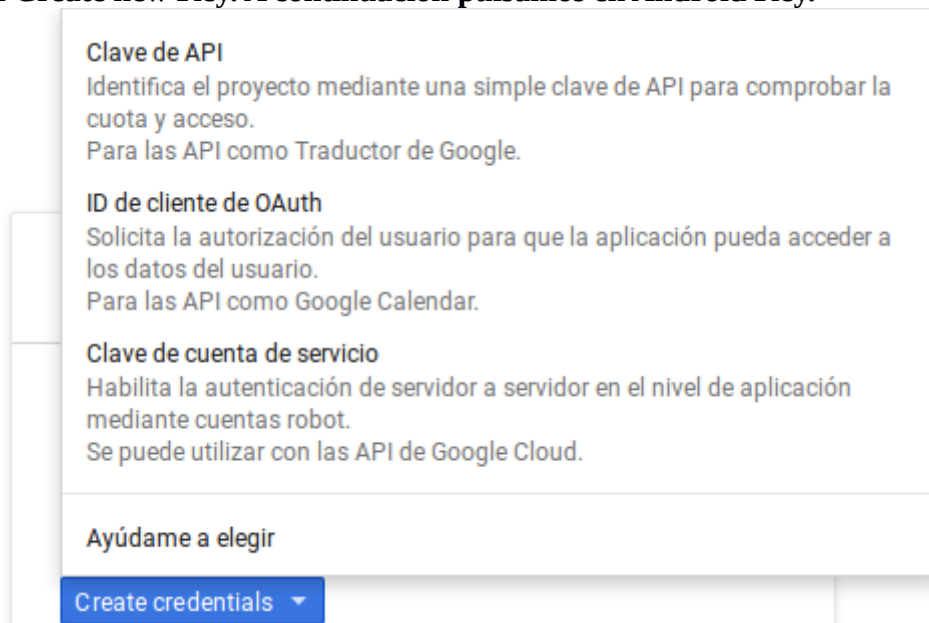
1. Google Maps API v2.
2. Google Maps Coordinate API.(funciona como Google Maps Direction)
3. Google Maps Engine API. (reemplazado por Google Maps Android API)
4. Google Maps Geolocation API.
5. Translate API.

Damos click en Usar APIs de Google, lista de APIs.

API ^

BigQuery API	Inhabilitar
Cloud Debugger API	Inhabilitar
Debuglet Controller API	Inhabilitar
Google Cloud Logging API	Inhabilitar
Google Cloud SQL	Inhabilitar
Google Cloud Storage	Inhabilitar
Google Cloud Storage JSON API	Inhabilitar
Google Maps Android API	Inhabilitar
Google Maps Directions API	Inhabilitar
Google Maps Geolocation API	Inhabilitar

3. Una vez activadas las APIs pulsamos sobre Credentials y sobre el botón Create new Key. A continuación pulsamos en Android Key.



Clave de API
Identifica el proyecto mediante una simple clave de API para comprobar la cuota y acceso.
Para las API como Traductor de Google.

ID de cliente de OAuth
Solicita la autorización del usuario para que la aplicación pueda acceder a los datos del usuario.
Para las API como Google Calendar.

Clave de cuenta de servicio
Habilita la autenticación de servidor a servidor en el nivel de aplicación mediante cuentas robot.
Se puede utilizar con las API de Google Cloud.

[Ayúdame a elegir](#)

Create credentials ▾

4. Una vez que tenemos nuestra clave tenemos que dejar constancia a Android de ello. Para este fin abrimos una consola y en la carpeta “.android” si estamos en Linux, introducimos el comando keytool. En nuestra consola escribimos “keytool -list -v -keystore debug.keystore” En mi máquina, no tiene clave así que seguimos

```
jbot@jlap:~/android$ keytool -list -v -keystore debug.keystore
Introduzca la contraseña del almacén de claves:

***** WARNING WARNING WARNING *****
* La integridad de la información almacenada en el almacén de claves *
* NO se ha comprobado. Para comprobar dicha integridad, *
* debe proporcionar la contraseña del almacén de claves. *
***** WARNING WARNING WARNING *****

Tipo de Almacén de Claves: JKS
Proveedor de Almacén de Claves: SUN

Su almacén de claves contiene 1 entrada

Nombre de Alias: androiddebugkey
Fecha de Creación: 13/01/2016
Tipo de Entrada: PrivateKeyEntry
Longitud de la Cadena de Certificado: 1
Certificado[1]:
Propietario: CN=Android Debug, O=Android, C=US
Emisor: CN=Android Debug, O=Android, C=US
Número de serie: 78bc40ce
Válido desde: Wed Jan 13 06:47:23 PET 2016 hasta: Fri Jan 05 06:47:23 PET 2046
Huellas digitales del Certificado:
    MD5: 17:5B:5D:B0:25:A5:CC:11:35:D7:1E:60:F5:BD:F4:EA
    SHA1: 8D:27:E2:D1:78:C5:45:F5:6E:EC:94:31:09:EE:BB:CB:4D:D3:CD:7F
    SHA256: 45:4C:65:18:1B:31:81:DC:9E:BC:17:7E:52:E6:36:1B:8E:F6:7B:B8:2C:DE:A8:89:D1:2F:E9:FB:8D:45:8B:46
Nombre del Algoritmo de Firma: SHA256withRSA
Versión: 3

Extensiones:
#1: ObjectId: 2.5.29.14 Criticality=false
SubjectKeyIdentifier [
KeyIdentifier [
0000: E2 0B 8F E2 3A B0 9D BF 44 C3 6E A3 E8 45 DC FB .....D.n..E..
0010: 10 2C B3 C2 .....
]
]
```

Copiamos el SHA1 en el campo que nos pide, junto al nombre de nuestro paquete.
Y Le damos en guardar (debido a que ya había creado una rápida y ahora recién estoy modificando).

Clave de API de Android

Clave de API	AlzaSyAob7mOamfDNf4wAWu-SHhtAyCBs3M0_pA
Fecha de creación	4 mar. 2016 14:58:30
Creada por	felipe.moreno.vera@gmail.com (tú)

Nombre

AndroidKey-1

Restringir el uso a tus aplicaciones Android (Opcional)

Los dispositivos Android envían solicitudes de API directamente a Google. Google verifica que cada una de las solicitudes proceda de una aplicación para Android que coincida con un nombre de paquete y una huella digital de certificado de firma SHA-1 que nos proporciones. Puedes encontrar el nombre de paquete en el archivo AndroidManifest.xml. Usa el comando siguiente para obtener la huella digital. [Más información](#)

```
keytool -list -v -keystore mystore.keystore
```

Nombre de paquete

Huella digital de certificado SHA-1

com.example.jbot.migooglemapsapi

8D:27:E2:D1:78:C5:45:F5:6E:EC:94:31:09:EE:BB:CB:4D:D3:CD:7F

×

5. Una vez introducida la contraseña y pulsado Intro copiamos la huella en la ventana del navegador. Además introducimos el nombre de nuestro paquete (en mi caso com.example.manwest.maps) separado por “;”

Nos generó nuestra android key como se ve arriba.

6. Una vez copiada nuestra API Key deberemos copiarla en nuestro AndroidManifest.xml.

<meta-data

android:name="com.google.android.geo.API_KEY"

android:value="@string/google_maps_key" />

Aquí el string google_maps_key debe tener la Key generada en la web

Va dentro de tu etiqueta <application en el manifest

7. Por otro lado en el mismo Manifiesto deberemos establecer que la API v2 de Google Maps Android utiliza OpenGL ES versión 2

<uses-feature

android:glEsVersion="0x00020000"

android:required="true"/>

3. El layout de nuestra actividad.

```
<fragment xmlns:android="http://schemas.android.com/apk/res/android"
xmlns:tools="http://schemas.android.com/tools"
android:layout_width="match_parent"
android:layout_height="match_parent"
android:id="@+id/map"
tools:context=".MapsActivity"
android:name="com.google.android.gms.maps.SupportMapFragment" />
```

android:name debe ser el nombre de tu proyecto referenciando al nombre de tu activity main.

Genera errores que solucione con [1].

4. Nuestra actividad principal al utilizar Mapas como si fuera un Fragment descenderá de FragmentActivity.

```
public class MainActivity extends FragmentActivity {
```

5. Nuestro onCreate lo vamos a estructurar llamando al método setUpMapIfNeeded.

```
@Override
protected void onCreate(Bundle savedInstanceState) {
    super.onCreate(savedInstanceState);
    setContentView(R.layout.activity_main);
    setUpMapIfNeeded();
}
```

6. Luego el método onResume también llamará al método anterior, quedando de la siguiente manera.

```
@Override
protected void onResume() {
    super.onResume();
    setUpMapIfNeeded();
}
```

7. Por tanto el método setUpMapIfNeeded.

```
private void setUpMapIfNeeded() {
    if (mMap == null) {
        mMap = ((SupportMapFragment) getSupportFragmentManager()
            .findFragmentById(R.id.map)).getMap();
        if (mMap != null) {
            mMap.setMyLocationEnabled(true);
            setUpMap();
        }
    }
}
```

8. Lo primero que vamos a realizar es realizar una modificación del Marker que aparece por defecto, para ello nos vamos al método setUpMap y ponemos las coordenadas de Lima.

```
private void setUpMap() {
    nMap.addMarker(new MarkerOptions().position(new LatLng(-12.017124,
    -77.050744)).title("Facultad de Ciencias"));
}
```

9. Si ejecutamos la aplicación nos damos cuentas que no sale centrado en el punto que acabamos de señalar. Para que aparezca centrado:

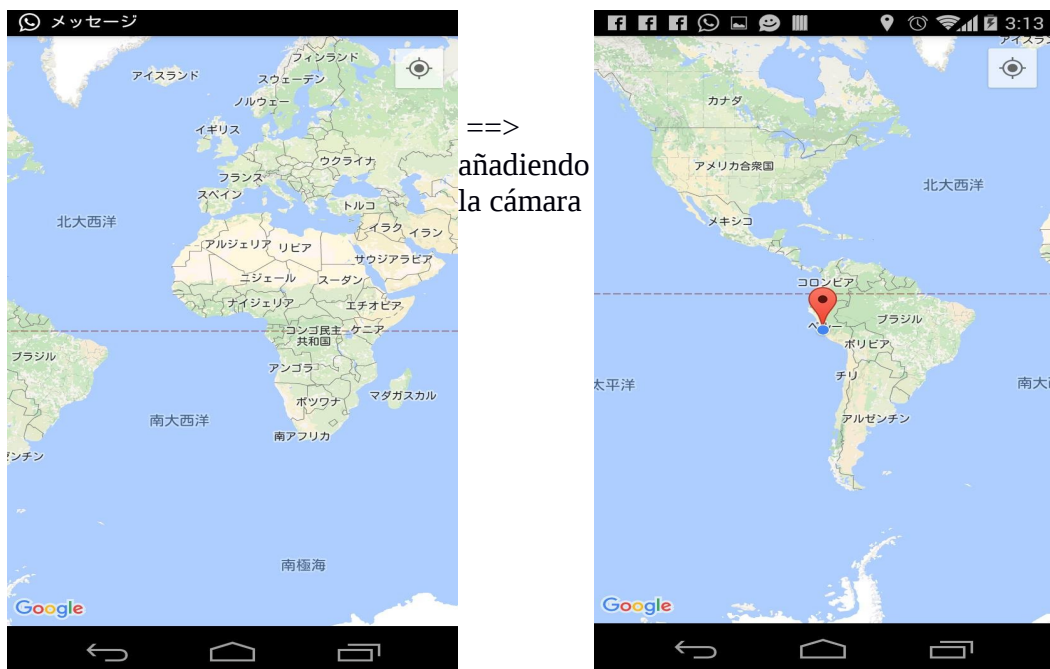
1. Primero declaramos una variable CameraUpdate importando también la librería correspondiente.

```
private CameraUpdate mCamera;
```

2. Una vez creada, le decimos que se centre en nuestro Marker al abrir la aplicación. Para realizarlo de forma animada usaremos animateCamera modificando el método setUpMap.

```
private void setUpMap() {
    nMap.addMarker(new MarkerOptions().position(new LatLng(-12.017124,
    -77.050744)).title("Facultad de Ciencias"));
    nCamera = CameraUpdateFactory.newLatLngZoom(new LatLng(
    -12.017124, -77.050744), 0);
    nMap.animateCamera(mCamera);
}
```

Solución hasta ahora según la referencia de [2] y [3].



10. Ahora vamos a modificar el color del Marker y añadir una descripción al mismo.

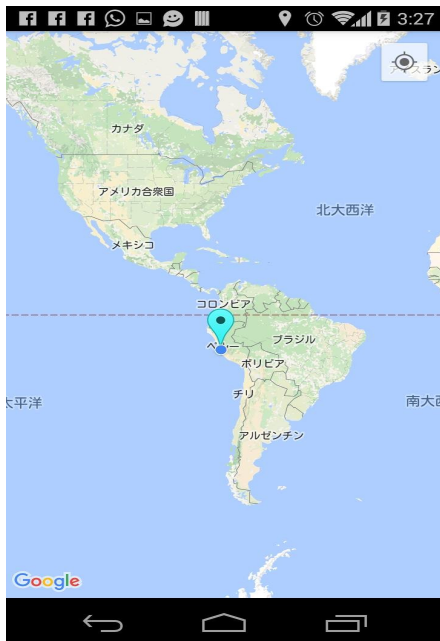
1. Para modificar el color por ejemplo será la opción .icon y la descripción con .snippet.

```
private void setUpMap() {  
    nMap.addMarker(new MarkerOptions().position(new LatLng(-12.017124,  
-77.050744)).title("Facultad de Ciencias")  
    .icon(BitmapDescriptorFactory.defaultMarker(  
    BitmapDescriptorFactory.HUE_CYAN))  
    .snippet("The beast School");  
    nCamera = CameraUpdateFactory.newLatLngZoom(new LatLng(  
-12.017124, -77.050744), 0);  
    nMap.animateCamera(mCamera);  
}
```

11. Ejecutamos y vemos el resultado. Cambie el valor de CameraUpdate de 0 por el valor 14 y explique que ha sucedido. También hay muchas más opciones, del enlace que se expone a continuación explicar dos opciones más.

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

Cambió el color del icono(izquierda) y al cambiarle de 0 a 14, le dio un zoom de 14(derecha).



Anchor

El punto en la imagen que se posiciona el la longitud y latitud, por default esta en el medio.

Alpha

Configura el nivel de opacidad del marker, por default es 1.0.

12. Vamos a cambiar los iconos de los Markers mediante el método addMarker pero esta vez desde fromResource. Para ello coge las imágenes que se han dejado en la práctica y súbelas a la carpeta drawable. Una vez cambiado vemos el resultado
`.icon(BitmapDescriptorFactory.fromResource(R.drawable.cafeteria))`



13. Bien vamos a comentar la línea que añade nuestro Marker (mMap.addMarker...) dejando solamente mCamera y mMap, animateCamera del método setUpMap. Ahora vamos a personalizar nuestros Markers mediante un método. Explica cada propiedad.

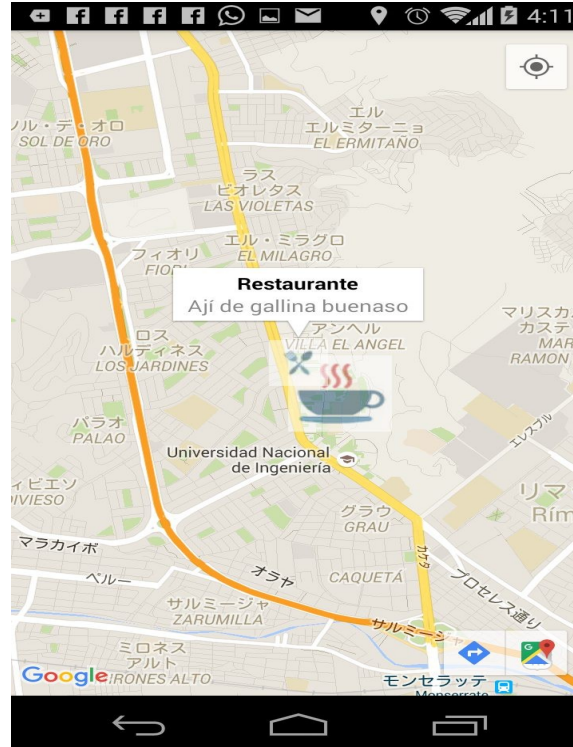
```
private void setMarker(LatLng position, String title, String info, float opacity, float
dimension1, float dimension2, int icon){
mMap.addMarker(new MarkerOptions()
.position(position)           // posición
.title(title)                 // título al dar click
.snippet(info)                // información al dar click
.alpha(opacity)               // opacidad
.anchor(dimension1, dimension2) // dimensión
.icon(BitmapDescriptorFactory.fromResource(icon))); // icono }
```

14. Una vez creados nuestro estilo de Markers desde el onCreate después del método setUpMapIfNeeded vamos a dibujarlos.

```
setMarker(new LatLng( -12.017128, -77.050748 ), "Cafetería", "El mejor café", 0.9F, 0.1F,
0.1F, R.drawable.cafeteria);
setMarker(new LatLng( -12.017124, -77.050744 ), "Restaurante", "Ají de gallina buenaso",
0.5F, 0.5F, 0.5F, R.drawable.restaurante);
```

15. Ejecute y vea la salida.

Solución hasta ahora, se observa ambos puntos y además al dar click nos muestra la info del punto restaurante,



16. Por último vamos a añadir la visualización de los diferentes mapas que nos da Android para ver.

1. Establecemos por defecto el mapa Híbrido. Añadimos la siguiente línea en negrita.

```
mMap.setMyLocationEnabled(true);  
mMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);  
setUpMap();
```

2. Ahora vamos a añadir al menú las diferentes opciones.

```
public boolean onOptionsItemSelected(MenuItem item) {  
    switch (item.getItemId()) {  
        case R.id.MenuOpcion1:  
            mMap.setMapType(GoogleMap.MAP_TYPE_NORMAL);  
            return true;  
        case R.id.MenuOpcion2:  
            mMap.setMapType(GoogleMap.MAP_TYPE_SATELLITE);  
            return true;  
        case R.id.MenuOpcion3:  
            mMap.setMapType(GoogleMap.MAP_TYPE_TERRAIN);  
            return true;  
    }  
}
```

```

case R.id.MenuOpcion4:
mMap.setMapType(GoogleMap.MAP_TYPE_HYBRID);
return true;
default:
return super.onOptionsItemSelected(item);
} }

```

3. El archivo XML del menú correspondiente.

```

<menu xmlns:android="http://schemas.android.com/apk/res/android" >
<item android:id="@+id/MenuOpcion1"
    android:title="Normal" />
<item android:id="@+id/MenuOpcion2"
    android:title="Satélite" />
<item android:id="@+id/MenuOpcion3"
    android:title="Terreno" />
<item android:id="@+id/MenuOpcion4"
    android:title="Híbrido" />
</menu>

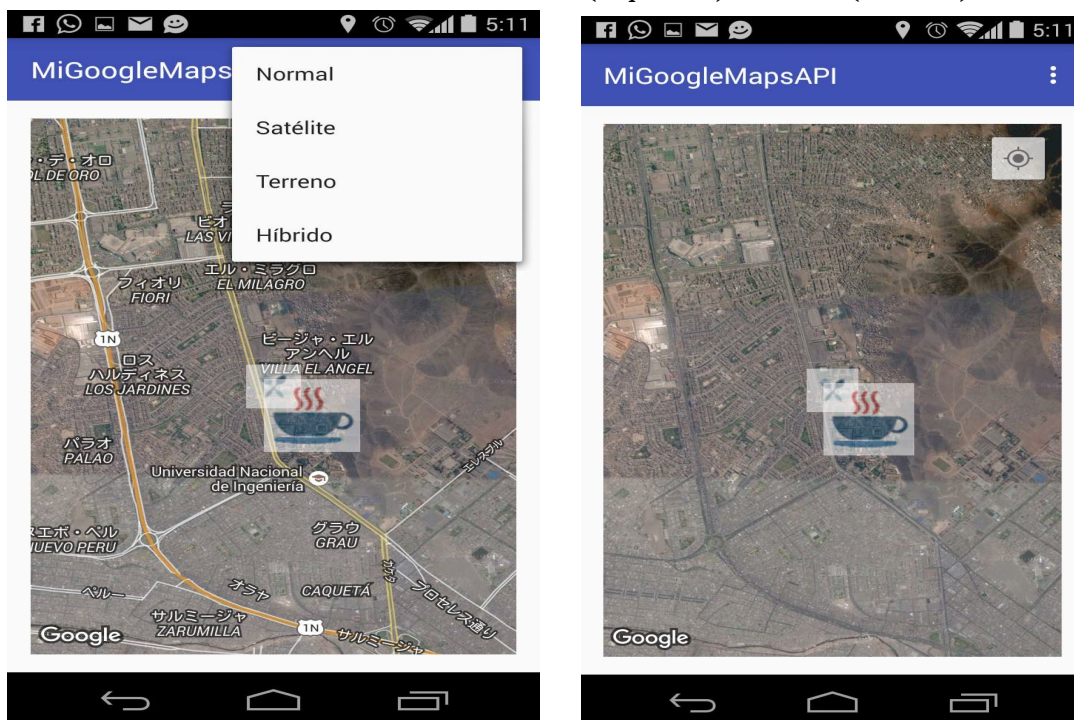
```

Hay que añadir `app:showAsAction="never"` a cada item.

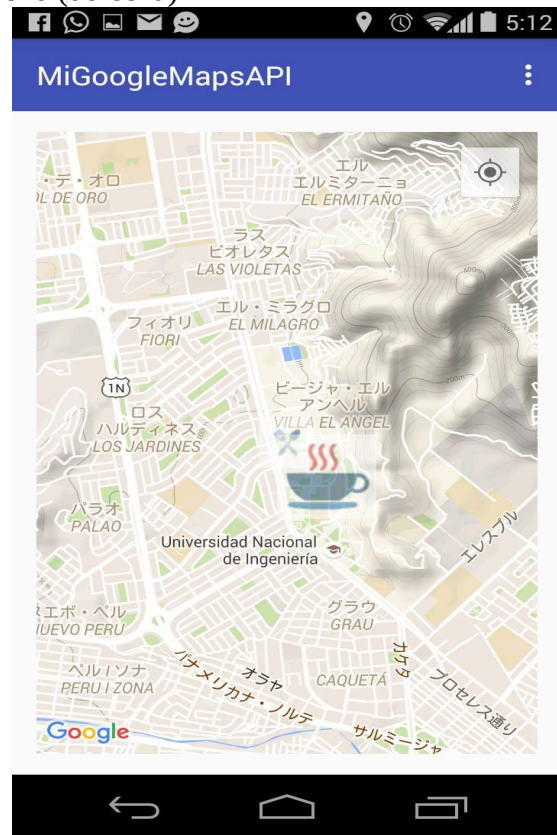
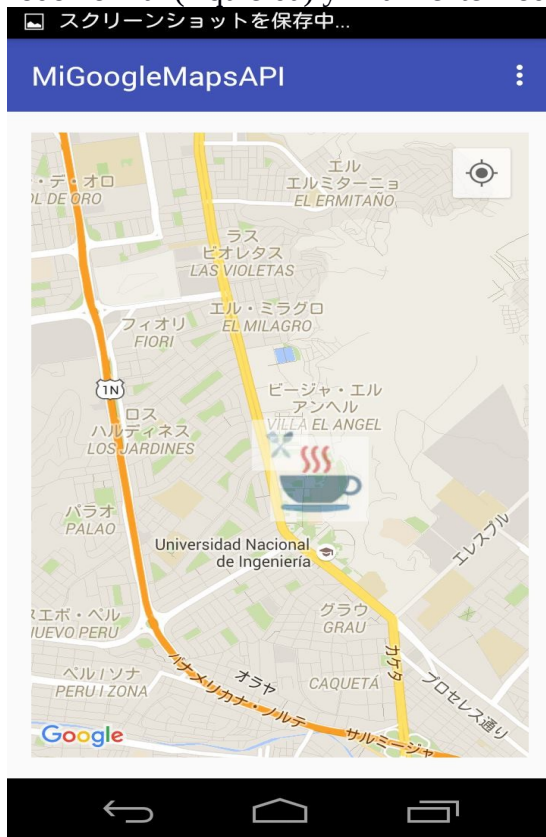
17. Ejecute de nuevo la aplicación y vea el resultado de cada uno.

Para lograr que se muestre el Toolbar, tenemos que cambiar de `extends FragmentActivity` a `AppCompatActivity`.

Ahora con las 4 modos, tenemos: modo híbrido(izquierda) , satélite (derecha).



modo normal (izquierda) y finalmente modo terreno (derecha)



18. Hay otros aspectos que permiten personalizar mapas. Se pide exponer cada uno de ellos.

1. Crear rectas y polígonos.

Se crea mediante una clase aparte llamada Polígono que toma como puntos coordenadas en el mapa.

```
public class Posiciones { //Nueva clase Posiciones
    //Constante de Posición del marcador
    public static final LatLng SAGRADA_FAMILIA = new LatLng(41.40347,
        2.17432);

    //Constante de Opciones de Polilínea.
    public static final PolylineOptions POLILINEA = new PolylineOptions()
        .add(new LatLng(41.40347, 2.17432))
        .add(new LatLng(41.40691, 2.16864))
        .add(new LatLng(41.40364, 2.16437));
}
```

Creamos un método que dibuje el polígono.

```
private void drawPolilyne(PolylineOptions options){  
    Polyline polyline = mMap.addPolyline(options);  
}
```

Y finalmente lo añadimos al onCreate:

```
setMarker(Posiciones.SAGRADA_FAMILIA,"Sagrada Familia","Distrito:  
Barcelona");  
drawPolilyne(Posiciones.POLILINEA);
```

2. Cambiar orientación, ángulo y zoom de la cámara.

```
// mover la camara instantaneamente a Sydney con un zoom  
map.moveCamera(CameraUpdateFactory.newLatLngZoom(SYDNEY, 15));
```

```
// agrandamos el zoom en 1,animando la camara  
map.animateCamera(CameraUpdateFactory.zoomIn());
```

```
// alejamos el zoom al nivel 10, animando la cámara por 2 segundos  
map.animateCamera(CameraUpdateFactory.zoomTo(10), null, 2000);
```

```
// Construimos una camara que apunta a Mountain View y animamos la camara hasta esa  
// posicion
```

```
CameraPosition cameraPosition = new CameraPosition.Builder()  
    .target(MOUNTAIN_VIEW) // setemos el centro del mapa a Mountain View  
    .zoom(17) configuramos el zoom  
    .bearing(90) // cambiamos la orientación de la cámara al este  
    .tilt(30) // configuramos el angulo de la cámara a 30 grados  
    .build(); // Creamos la CameraPosition
```

```
map.animateCamera(CameraUpdateFactory.newCameraPosition(cameraPosition));
```

3. Listeners de setOnMarkerClickListener, setOnCameraChangeListener, setOnMapLongClickListener,

* OnInfoWindowClickListener se activa cuando el usuario hace clic en la ventana de información que aparece en un marcador en el mapa.

* OnCameraChangeListener es llamada después de que la posición de la cámara ha cambiado. Durante una animación, este listener no puede ser notificado de posiciones de las cámaras intermedias. Siempre es llamado para la posición final en la animación.

* OnMapLongClickListener y OnMapClickListener se activan cuando el usuario ya sea grifos o mantiene abajo en una porción del mapa.

* OnMarkerClickListener se llama cuando el usuario hace clic en un marcador en el mapa, que por lo general también muestra la ventana de información para ese marcador.

Referencia en [4]

Link del github con los códigos del laboratorio:

https://github.com/Jenazad/PDM/tree/master/Laboratorio_10

Referencias

[1]

<http://stackoverflow.com/questions/13726189/android-google-maps-api-v2-supportmapfragment-errors>

[https://developers.google.com/android/guides/setup#add google play services to your project](https://developers.google.com/android/guides/setup#add_google_play_services_to_your_project)

<http://stackoverflow.com/questions/26901149/com-google-android-gms-maps-mapfragment-cannot-resolve-symbol-maps>

[2]

<http://expocodetech.com/usar-google-maps-en-aplicaciones-android-mapa/>

<http://stackoverflow.com/questions/19353255/how-to-put-google-maps-v2-on-a-fragment-using-viewpager>

[3]

<http://code.tutsplus.com/tutorials/getting-started-with-google-maps-for-android-basics--cms-24635>

<https://github.com/treehouse/android-location-example/blob/master/app/src/main/java/teamtreehouse/com/iamhere/MapsActivity.java>

[4]

<https://developers.google.com/android/reference/com/google/android/gms/maps/GoogleMap.OnCameraChangeListener#public-method-summary>

<http://www.aprendiendodeandroidymas.com/2013/04/comenzando-con-google-maps-api-v2-parte.html>