Introducción a la Computación Móvil
Mapas (Google, OSM)

Pontificia Universidad Javeriana
Departamento de Ingeniería de Sistemas
Profesor: Carlos Andrés Parra
E-mail: ca.parraa@javeriana.edu.co

Mapas

Google Maps

Actividad para mapas
New Activity -> Gallery

- Se utiliza el asistente de creación de actividades
- Una vez la actividad ha sido creada, el asistente agrega las dependencias en el archivo gradle del módulo

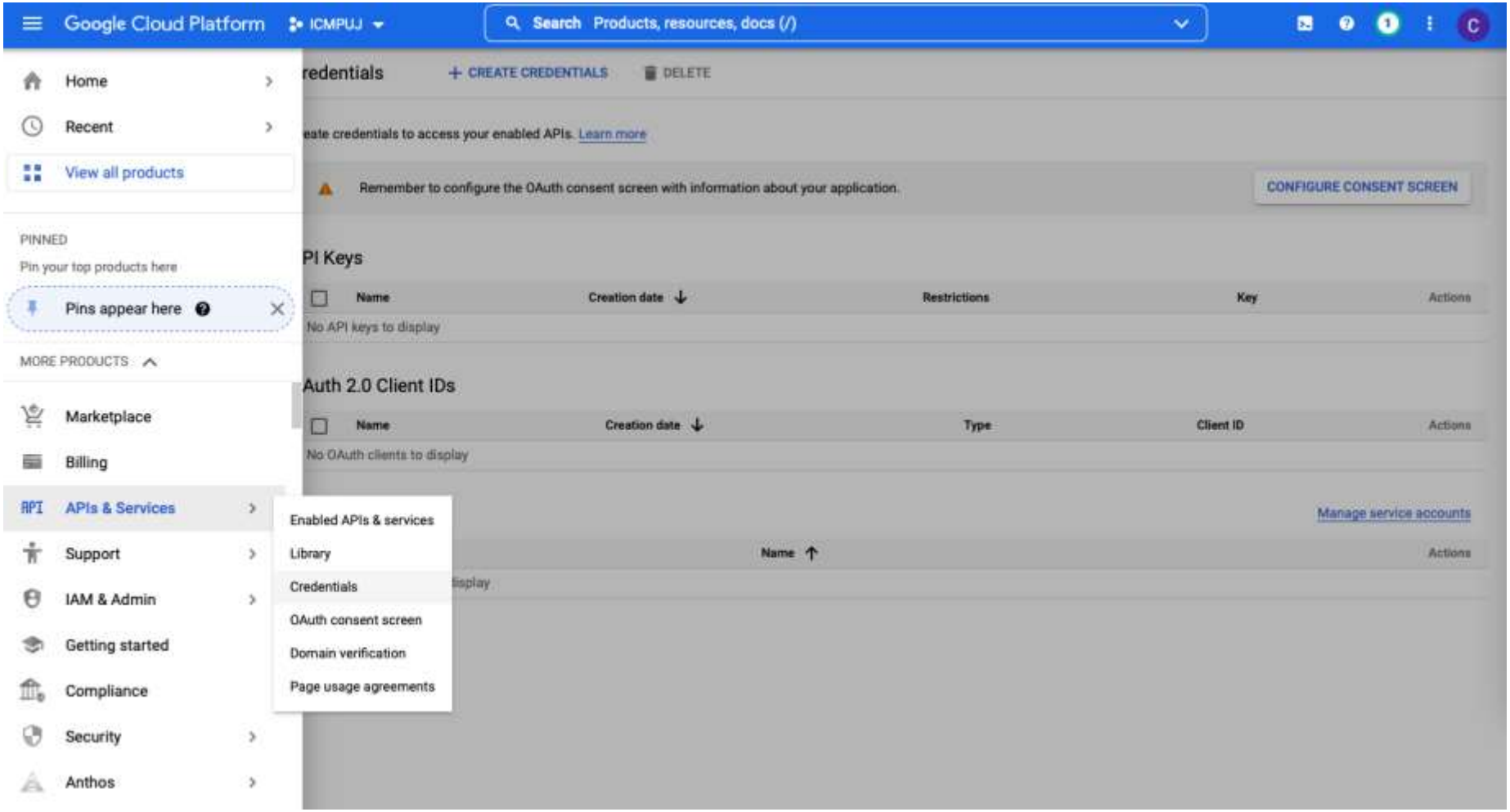

```
implementation 'com.google.android.gms:play-services-maps:18.0.2'
```

- Se crean tres archivos
  - La actividad y el layout del mapa
  - Un nuevo tag en el manifest con instrucciones para obtener un API KEY

Google Maps API Key
google_maps_api.xml

- Entrar a console.cloud.google.com
- Autenticarse con el correo de gmail o la cuenta que se quiera usar
- Crear o seleccionar un proyecto nuevo
- Ir al menu y seleccionar APIS and Services -> Library, entrar a la opción de mapas para Android y habilitarla
- Entrar a Apis and Services -> Credentials
  - Crear un API Key
  - opcionalmente restringirla (Sólo para aplicaciones Android, solo para Mapas, etc)

Api Key


Google Cloud Platform
ICMPUJ
Search Products, resources, docs (/)
Home
Recent
View all products
PINNED
Pin your top products here
Pins appear here
MORE PRODUCTS
Marketplace
Billing
APIs & Services
Support
IAM & Admin
Getting started
Compliance
Security
Anthos
Enabled APIs & services
Library
Credentials
OAuth consent screen
Domain verification
Page usage agreements
redentials
+ CREATE CREDENTIALS
DELETE
eate credentials to access your enabled APIs. Learn more
Remember to configure the OAuth consent screen with information about your application.
CONFIGURE CONSENT SCREEN
PI Keys
Name
Creation date
Restrictions
Key
Actions
No API keys to display
Auth 2.0 Client IDs
Name
Creation date
Type
Client ID
Actions
No OAuth clients to display
Manage service accounts
Name
Actions

API KEY Secret

- La clave no debería incluirse en los elementos que se envían al repositorio, para esto se puede usar el archivo local.properties en la base del proyecto:

```
## This file must *NOT* be checked into Version Control Systems,
# as it contains information specific to your local configuration.
#
# Location of the SDK. This is only used by Gradle.
# For customization when using a Version Control System, please
read the
# header note.
#Tue Mar 01 15:49:43 COT 2022
sdk.dir=/Users/andrew/Library/Android/sdk
MAPS_API_KEY=Aiza……
```

API KEY Secret

- En el archivo manifest, se hace referencia a la llave secreta

```
<!--
    TODO: Before you run your application, you need a Google Maps API key.

    To get one, follow the directions here:

       https://developers.google.com/maps/documentation/android-sdk/get-api-key

    Once you have your API key (it starts with "AIza"), define a new property in your
    project's local.properties file (e.g. MAPS_API_KEY=Aiza...), and replace the
    "YOUR_API_KEY" string in this file with "${MAPS_API_KEY}".
-->
<meta-data
    android:name="com.google.android.geo.API_KEY"
    android:value="${MAPS_API_KEY}"/>
```

Actividad de mapas

```
public class GoogleMapsActivity extends FragmentActivity implements OnMapReadyCallback {

    private GoogleMap mMap;
    private ActivityGoogleMapsBinding binding;

    @Override
    protected void onCreate(Bundle savedInstanceState) {
        super.onCreate(savedInstanceState);
        binding = ActivityGoogleMapsBinding.inflate(getLayoutInflater());
        setContentView(binding.getRoot());
        // Obtain the SupportMapFragment and get notified when the map is ready to be used.
        SupportMapFragment mapFragment = (SupportMapFragment) getSupportFragmentManager()
                .findFragmentById(R.id.map);
        mapFragment.getMapAsync(this);
    }

    @Override
    public void onMapReady(GoogleMap googleMap) {
        mMap = googleMap;

        // Add a marker in Sydney and move the camera
        LatLng bogota = new LatLng(4.6269938175930525, -74.06389749953162);
        mMap.addMarker(new MarkerOptions().position(bogota).title("Marker in Bogota"));
        mMap.moveCamera(CameraUpdateFactory.newLatLng(bogota));
    }
}
```

Actividad con Mapa

Niveles de Zoom
1: Mundo
5: Tierra firme y continente
10: Ciudad
15: Calles
20: Edificios

```
mMap.moveCamera(CameraUpdateFactory.zoomTo(10));
```

10

15

Interfaz de usuario para el mapa

- Controles del mapa:

  - Habilitar los “gestures” como “pinch to zoom”

```
mMap.getUiSettings().setZoomGesturesEnabled(true);
```

  - Habilitar los botones de zoom

```
mMap.getUiSettings().setZoomControlsEnabled(true);
```

  - Inclinación, rotación, desplazamiento etc.

Fragmentos

- Representa una parte de la interfaz o del comportamiento dentro de una actividad.
- Se pueden usar varios fragmentos en una actividad y reutilizarlos en varias actividades.
- Siempre debe estar integrado en una actividad y su ciclo de vida es propio pero depende del ciclo de vida de la actividad.
- Si la actividad esta en onPause() o onDestroy() los fragmentos igual, sin embargo cuando la actividad esta en onResume() cada fragmento se maneja de forma independiente.

Fragmentos

- El fragmento puede estar contenido dentro de la actividad o puede estar en un archivo diferente.

```
<LinearLayout
    android:layout_height="match_parent"
    android:layout_width="match_parent"
    android:orientation="vertical"
    xmlns:android="http://schemas.android.com/apk/res/android"
    android:padding="15dp">
    <EditText android:id="@+id/texto"
        android:layout_height="wrap_content"
        android:layout_width="match_parent"
        android:hint="@string/hintAddress"/>
    <fragment
        android:id="@+id/map"
        android:name="com.google.android.gms.maps.SupportMapFragment"
        android:layout_width="match_parent"
        android:layout_height="match_parent"/>
</LinearLayout>
```

FrameLayout!

- Con un frameLayout, el mapa puede permanecer del mismo tamaño y el cuadro de texto puede aparecer encima con un nivel de transparencia.
- Para la transparencia, puede definir los dos primeros valores del color de tamaños menores para algo más transparente, y mayores para algo menos transparente

  "#bb0099cc"

- Y el cuadro de texto con bordes redondeados?

Rounded Corners

- En drawable, se puede crear un archivo ”roundcorner.xml” (todo en minúsculas):

```
<shape xmlns:android="http://schemas.android.com/apk/res/android"
   android:shape="rectangle" android:padding="10dp">
   <solid android:color="#bb0099cc"/>
   <corners
      android:bottomRightRadius="10dp"
      android:bottomLeftRadius="10dp"
      android:topLeftRadius="10dp"
      android:topRightRadius="10dp"/>
</shape>
```

Fondo semitransparente

- En el layout:

```
<EditText
      …
      android:background="@drawable/roundcorner”
      …
/>
```

Ejercicio 1

- Crear una aplicación básica que tenga un botón y al oprimirlo lance una actividad de mapas.
  - Utilice el asistente de nueva actividad y siga las instrucciones del archivo google_maps_api.xml para configurar una clave para el API de Google.
- Utilice google maps desde un navegador para obtener la latitud y longitud de la plaza de bolivar de Bogotá, e inicialice el mapa en esa posición con un nivel de zoom de 15.
- Habilite los “gestures” sobre el mapa.

Marcadores

- Son indicadores de una posición en el mapa. Se pueden personalizar de forma simple (colores) o incluso colocando un ícono personalizado.

```
LatLng bogota = new LatLng(4.65, -74.05);
mMap.addMarker(new MarkerOptions().position(bogota).title("Marcador en Bogotá"));
```

- Las ventanas de información asociadas proporcionan información adicional.
- Agregando un snippet se agrega información que se despliega cuando se hace click en el marcador

```
mMap.addMarker(new MarkerOptions().position(bogota)
      .title("BOGOTÁ")
      .snippet("Población: 8.081.000") //Texto de información
      .alpha(0.5f)); //Transparencia
```

Marcadores


11:28
Cogua
Nemocón
Zipaquirá
Tocancipá
Guatavita
Cajicá
Sopó
Tenjo
Guasca
Madrid
Mosquera
Bogotá
CHAPINERO
Monserrate
Soacha
Choachí
Sibaté
Fómeque
Chipaque
Fosca
Quetame
Pasca
La Calera
Ríoblanco
El Ramal
Guayabetal
Google


11:28
BOGOTÁ
Población: 8.081.000
Cogua
Nemocón
Zipaquirá
Tocancipá
Guatavita
Cajicá
Sopó
Tenjo
Guasca
Madrid
Mosquera
Bogotá
CHAPINERO
Monserrate
Soacha
Choachí
Sibaté
Fómeque
Chipaque
Fosca
Quetame
Pasca
La Calera
Ríoblanco
El Ramal
Guayabetal
Google

Marcadores Personalizados

- Se puede cambiar el color del marcador

```
Marker bogotaAzul = mMap.addMarker(new MarkerOptions()
      .position(bogota)
      .icon(BitmapDescriptorFactory
      .defaultMarker(BitmapDescriptorFactory.HUE_BLUE)));
```

- O utilizar una imagen personalizada

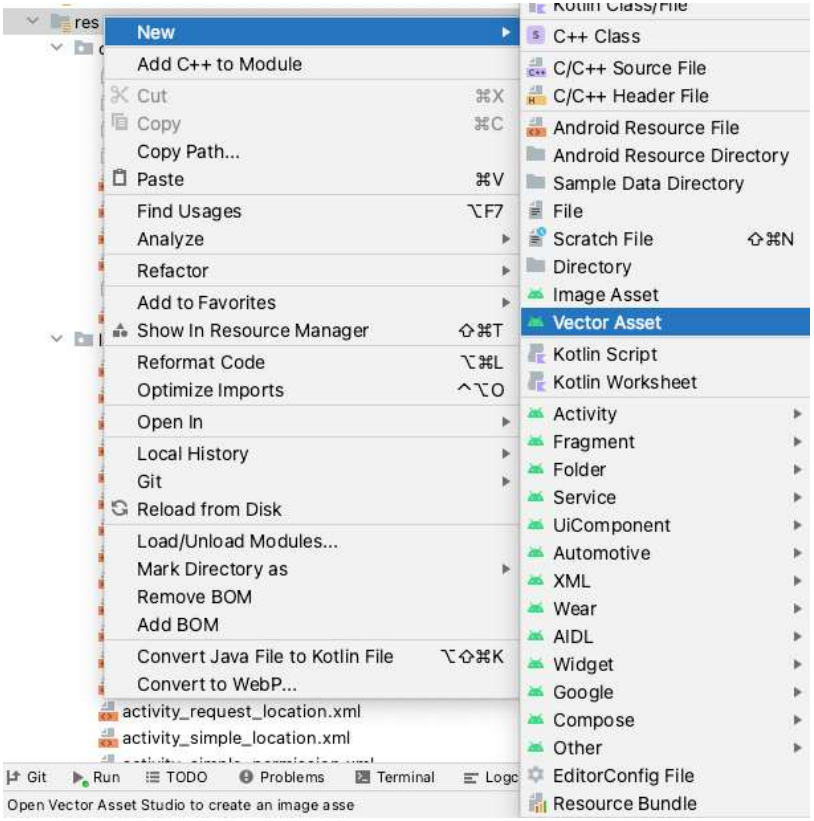
```
Marker bogotaBike = mMap.addMarker(new MarkerOptions()
      .position(bogota)
      .icon(BitmapDescriptorFactory
      .fromResource(R.drawable.bike)));
```

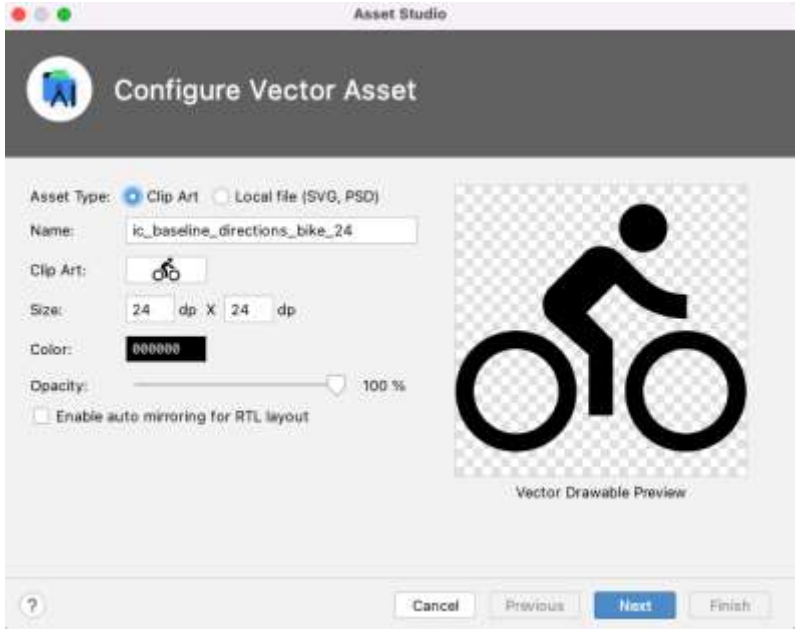
- Se pueden mostrar y ocultar los marcadores

```
bogotaAzul.setVisible(true);
bogotaAzul.setVisible(false);
```

Vector asset

- Android incluye una galería de íconos que se pueden agregar a los recursos de la aplicación (New->Vector Asset):





Usar el vector como marcador en el mapa

```
LatLng bogota2 = new LatLng(4.67, -74.07);
Marker bogotaBike = mMap.addMarker(new MarkerOptions().position(bogota2)
        .icon(bitmapDescriptorFromVector(this,R.drawable.ic_baseline_directions_bike_24)));
```

```
private BitmapDescriptor bitmapDescriptorFromVector(Context context, int vectorResId) {
    Drawable vectorDrawable = ContextCompat.getDrawable(context, vectorResId);
    vectorDrawable.setBounds(0, 0, vectorDrawable.getIntrinsicWidth(),
        vectorDrawable.getIntrinsicHeight());
    Bitmap bitmap = Bitmap.createBitmap(vectorDrawable.getIntrinsicWidth(),
        vectorDrawable.getIntrinsicHeight(), Bitmap.Config.ARGB_8888);
    Canvas canvas = new Canvas(bitmap);
    vectorDrawable.draw(canvas);
    return BitmapDescriptorFactory.fromBitmap(bitmap);
}
```

Usar el vector como marcador en el mapa

Quitar marcadores

- Cuando se hace seguimiento o hay mucha información, se hace necesario eliminar los marcadores.
  - Eliminar un sólo marcador:

```
Marker marker = mMap.addMarker(new MarkerOptions().position(latLng).title("Marcador"));
marker.remove();
```

Para poder eliminarlo se debe guardar en una variable

  - Limpiar el mapa (borra todos los marcadores):

```
mMap.clear();
```

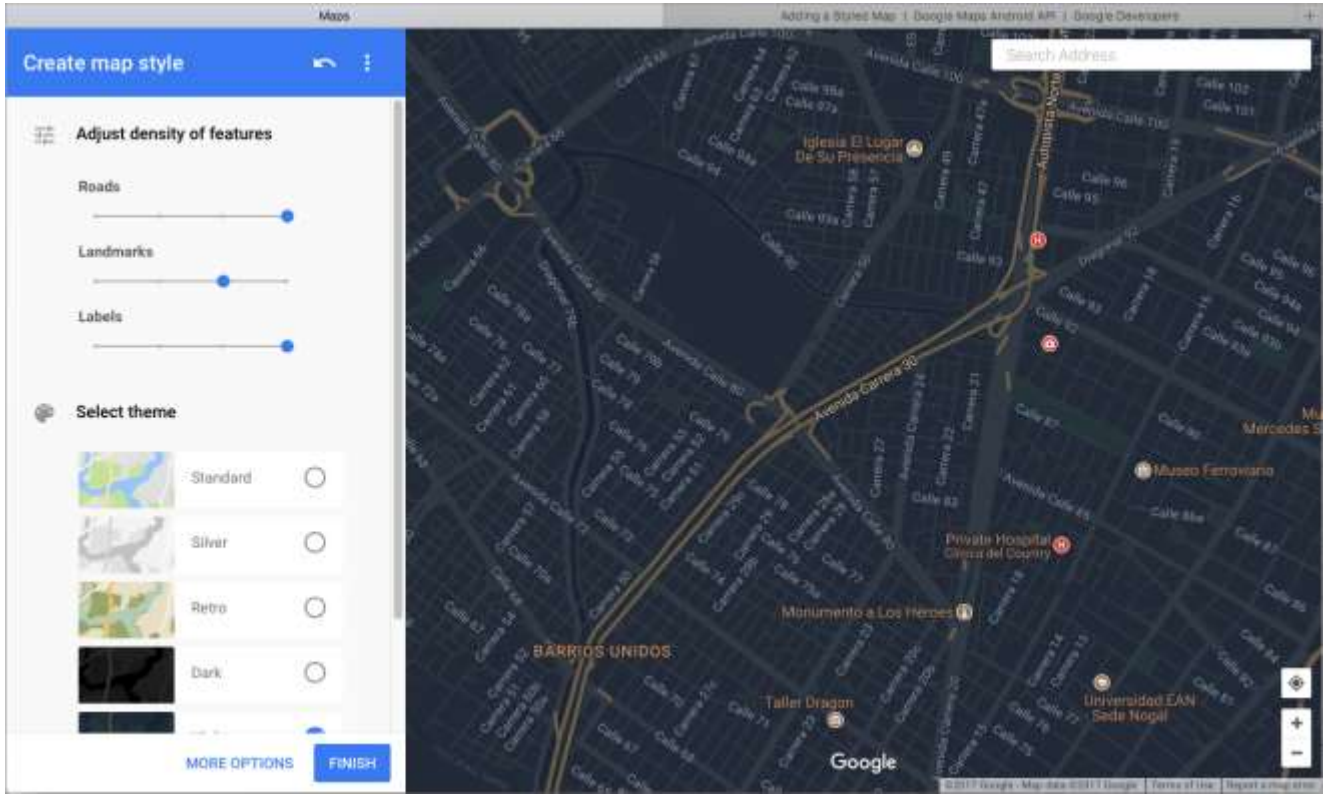
Ejercicio 2

- Con base en su aplicación de localización (código de la clase anterior) modifique el ejercicio 1 creando un marcador en la localización actual del usuario.
  - Utilice un marcador personalizado diferente al pin estándar de Google para la posición del usuario. Use una imagen de su preferencia.
- Agregue 5 marcadores para 5 lugares en Bogotá. Utilice marcadores propios y agregue información que caracterice a dichos lugares de interés. Esta información se debe mostrar al tocar cada marcador.

Personalizar mapas

- Es posible modificar la apariencia estándar de los mapas de google. Para esto, vaya al sitio:

https://mapstyle.withgoogle.com



Personalizar mapas

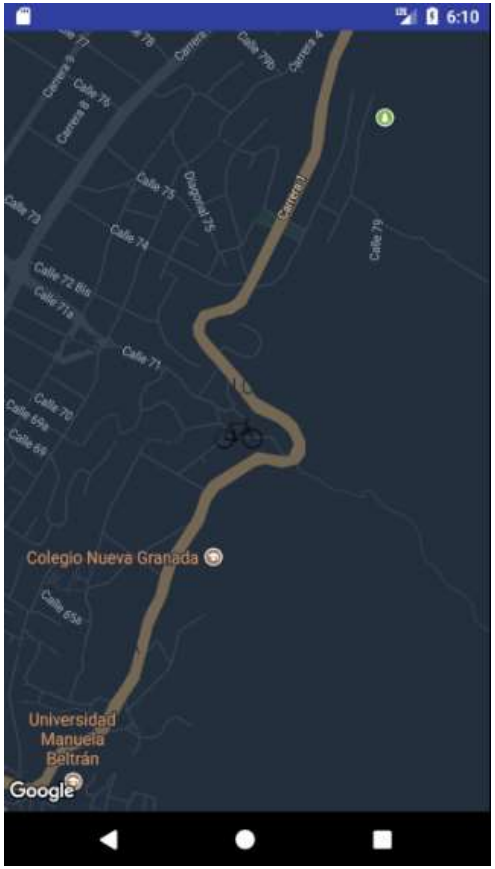
- El resultado de la personalización será un archivo JSON de propiedades del mapa, que se debe guardar en la siguiente ruta del proyecto (new Android Resource Directory -> Raw):

  /res/raw/style_json.json

- Desde el código, hay que hacer referencia al archivo de estilo del mapa:

```
mMap.setMapStyle(MapStyleOptions
        .loadRawResourceStyle(this,    R.raw.style_json));
```

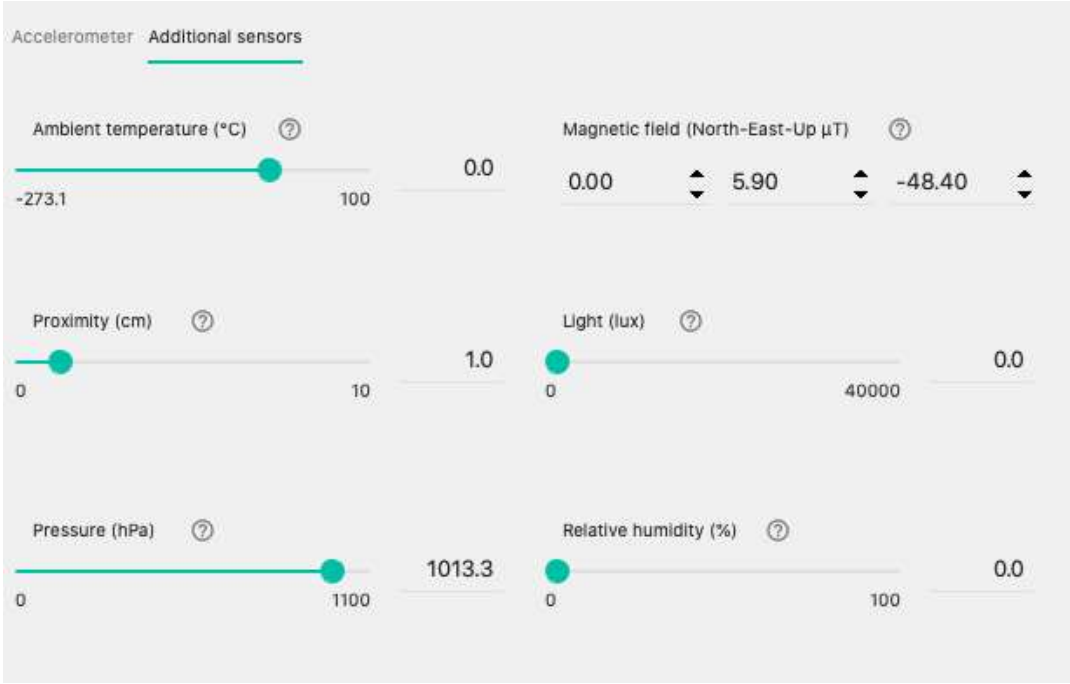
- Un caso de uso clásico es definir un estilo con colores oscuros en la noche. Para esto se pueden tener dos archivos de personalización del mapa y cambiarlos de acuerdo a la hora.



https://developers.google.com/maps/documentation/android-sdk/styling

Sensores y Hardware

- Android cuenta con un API que permite la consulta de los diferentes sensores del dispositivo. Estos sensores incluyen:
  - Temperatura
  - Presión Atmosférica
  - Luminosidad
  - Acelerómetro
  - Giroscopio
  - Proximidad



Uso del sensor de luminosidad

- Se necesitan tres atributos nuevos para la actividad disponibles en el paquete *android.hardware.**

```
SensorManager sensorManager;
Sensor lightSensor;
SensorEventListener lightSensorListener;
```

- Inicializar el administrador de los sensores

```
//onCreate
sensorManager = (SensorManager) getSystemService(SENSOR_SERVICE);
lightSensor = sensorManager.getDefaultSensor(Sensor.TYPE_LIGHT);
```

- El atributo lightSensor será nulo si el dispositivo no cuenta con este sensor.

Uso del sensor de luminosidad

- Definir las acciones del listener. El sensor de luminosidad recibe valores entre 0 y 40000 lx

```
lightSensorListener = new SensorEventListener() {
   @Override
   public void onSensorChanged(SensorEvent event) {
      if (mMap != null) {
         if (event.values[0] < 5000) {
            Log.i("MAPS", "DARK MAP " + event.values[0]);
            mMap.setMapStyle(MapStyleOptions.loadRawResourceStyle(MapsActivity.this, R.raw.dark_style_map));
         } else {
            Log.i("MAPS", "LIGHT MAP " + event.values[0]);
            mMap.setMapStyle(MapStyleOptions.loadRawResourceStyle(MapsActivity.this, R.raw.light_syle_map));
         }
      }
   }
   @Override
   public void onAccuracyChanged(Sensor sensor, int accuracy) {}
};
```

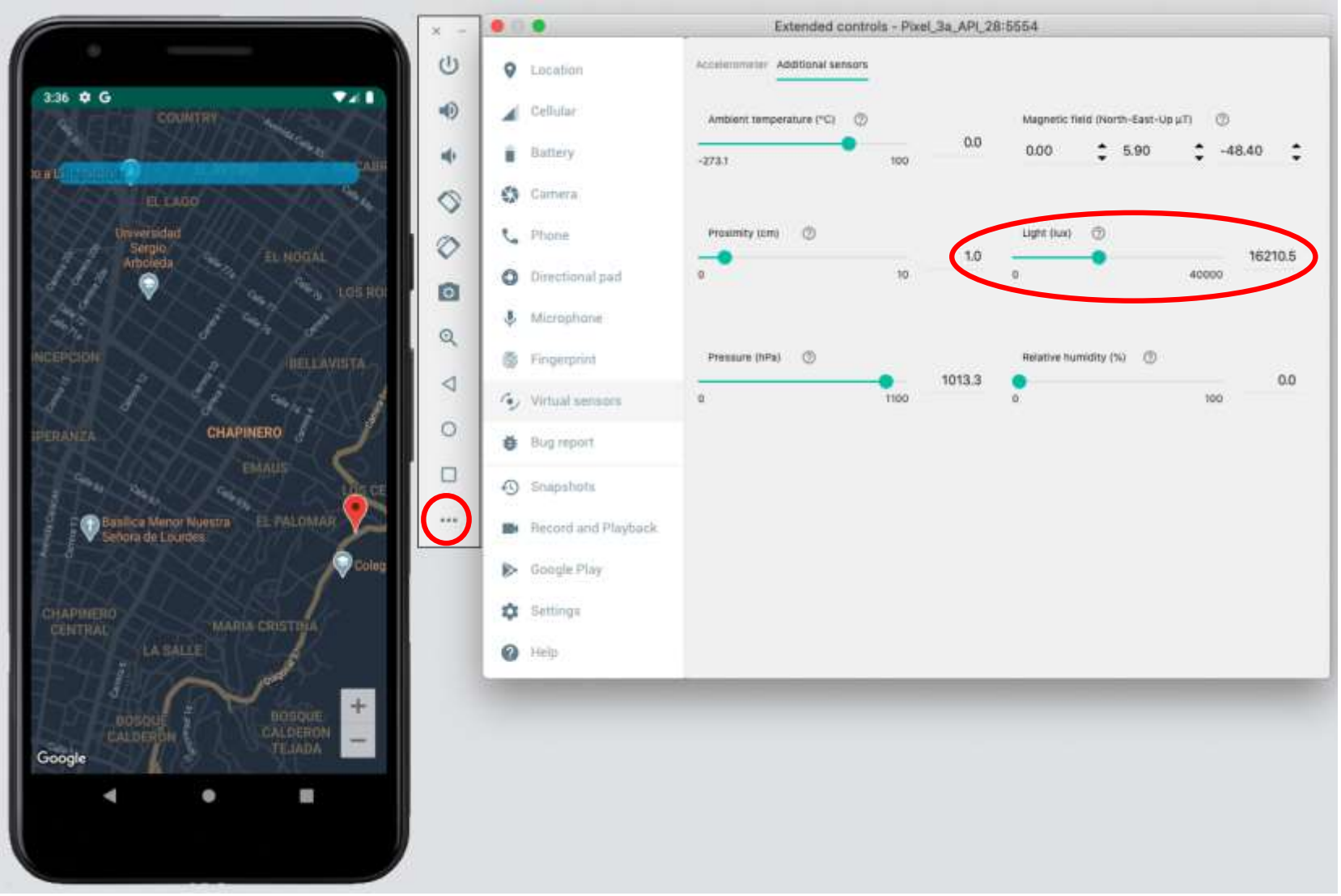
Registrarse a los eventos del sensor

- Al igual que con la localización, es importante suscribirse y desuscribirse de los cambios en el sensor usando los métodos adecuados del ciclo de vida de la actividad.

```
@Override
protected void onResume() {
   super.onResume();
   sensorManager.registerListener(lightSensorListener, lightSensor,
                                  SensorManager.SENSOR_DELAY_NORMAL);
}

@Override
protected void onPause() {
   super.onPause();
   sensorManager.unregisterListener(lightSensorListener);
}
```

Prueba en el emulador



API de Google para buscar direcciones

- Google provee un API para buscar direcciones de forma textual y obtener como resultado localizaciones que se pueden usar en un mapa.

Public methods	
`List<Address>`	`getFromLocation(double latitude, double longitude, int maxResults)` Returns an array of Addresses that are known to describe the area immediately surrounding the given latitude and longitude.
`List<Address>`	`getFromLocationName(String locationName, int maxResults, double lowerLeftLatitude, double lowerLeftLongitude, double upperRightLatitude, double upperRightLongitude)` Returns an array of Addresses that are known to describe the named location, which may be a place name such as "Dalvik, Iceland", an address such as "1600 Amphitheatre Parkway, Mountain View, CA", an airport code such as "SFO", etc..
`List<Address>`	`getFromLocationName(String locationName, int maxResults)` Returns an array of Addresses that are known to describe the named location, which may be a place name such as "Dalvik, Iceland", an address such as "1600 Amphitheatre Parkway, Mountain View, CA", an airport code such as "SFO", etc..
`static boolean`	`isPresent()` Returns true if the Geocoder methods getFromLocation and getFromLocationName are implemented.

https://developer.android.com/reference/android/location/Geocoder.html

API de Google para buscar direcciones
Listener de acciones en el teclado

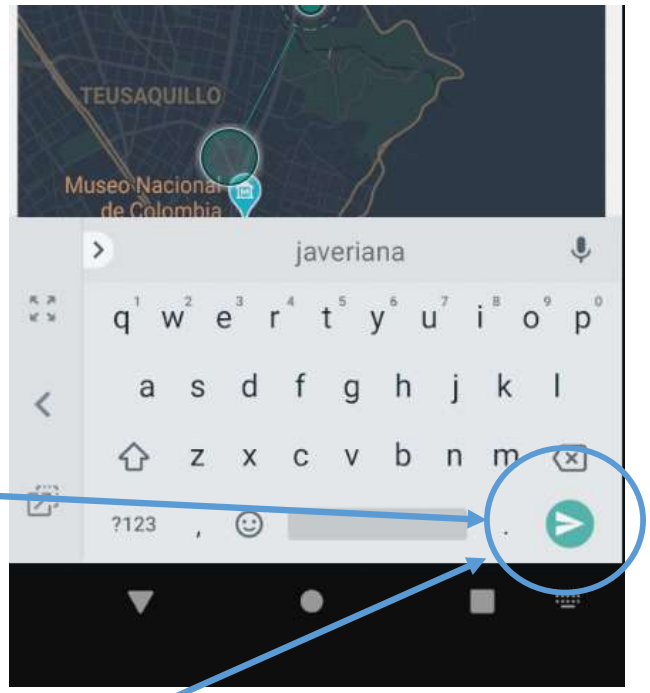
- Usar el objeto Geocoder

```
//Inicialización del objeto
Geocoder mGeocoder = new Geocoder(getBaseContext());

//Cuando se realice la busqueda
String addressString = mAddress.getText().toString();
if (!addressString.isEmpty()) {
      try {
            List<Address> addresses = mGeocoder.getFromLocationName(addressString, 2);
            if (addresses != null && !addresses.isEmpty()) {
                  Address addressResult = addresses.get(0);
                  LatLng position = new LatLng(addressResult.getLatitude(), addressResult.getLongitude());
                  if (mMap != null) {
                   //Agregar Marcador al mapa
                 }
            } else {Toast.makeText(MapsActivity.this, "Dirección no encontrada", Toast.LENGTH_SHORT).show();}
      } catch (IOException e) {
               e.printStackTrace();
          }
} else {Toast.makeText(MapsActivity.this, "La dirección esta vacía", Toast.LENGTH_SHORT).show();}
```

Actuar cuando el usuario termine de escribir

```
<EditText android:layout_width="match_parent"
   android:layout_height="46dp"
   android:id="@+id/searchMap"
   android:hint="Ingrese una dirección"
   android:inputType="text"
   android:imeOptions="actionSend”
   …
"/>
```




```
searchMap.setOnEditorActionListener(new TextView.OnEditorActionListener() {
   @Override
   public boolean onEditorAction(TextView v, int actionId, KeyEvent event) {
      if (actionId == EditorInfo.IME_ACTION_SEND){
         String addressString = searchMap.getText().toString();
         if (!addressString.isEmpty()) {
            try {
               List<Address> addresses = mGeocoder.getFromLocationName(addressString, 2);
               …
```

API de Google para buscar direcciones
Limitar búsqueda a una región

- Si se quiere limitar geográficamente la búsqueda a sólo una zona del planeta (por ejemplo Bogotá), se define una región a través de cuatro puntos:

```
//   Limits for the geocoder search  (Colombia)
    public static final double lowerLeftLatitude = 1.396967;
    public static final double lowerLeftLongitude= -78.903968;
    public static final double upperRightLatitude= 11.983639;
    public static final double upperRigthLongitude= -71.869905;
```

- Luego se invoca a geocoder incluyendo los parámetros:

```
List<Address> addresses = mGeocoder.getFromLocationName( addressString, 2,
        lowerLeftLatitude, lowerLeftLongitude,
        upperRightLatitude, upperRigthLongitude);
```

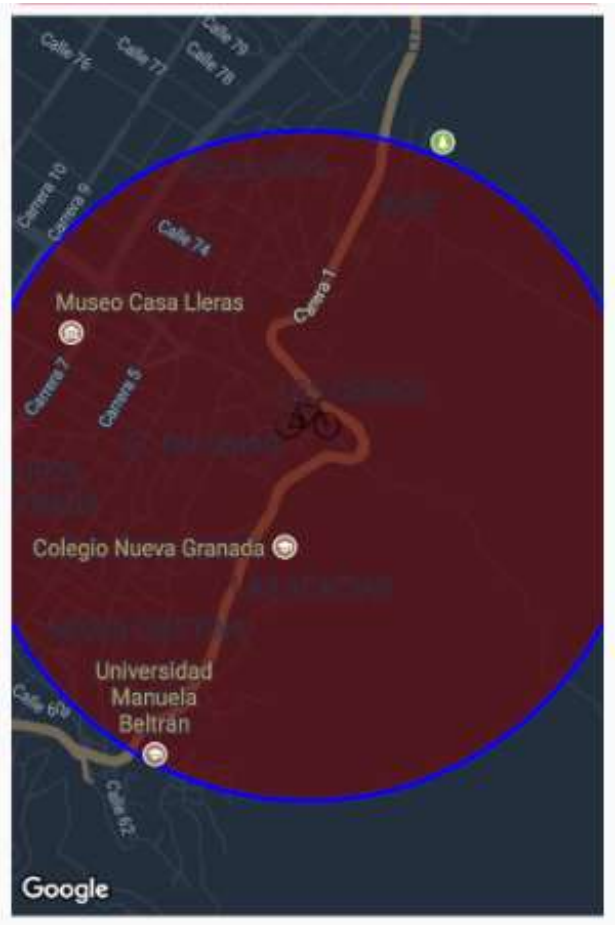
Listener sobre un mapa

- Es posible programar dos eventos en el mapa, Click o LongClick.
- El evento generado retorna las coordenadas (objeto LatLng) donde el usuario hace click.
- El siguiente fragmento de código genera un marcador con el nombre de la dirección encontrada donde la persona hace un click largo.


```
mMap.setOnMapLongClickListener(new GoogleMap.OnMapLongClickListener() {
   @Override
   public void onMapLongClick(LatLng latLng) {
      mMap.clear();
      mMap.addMarker(new MarkerOptions().position(latLng).title(geoCoderSearchLatLang(latLng)));
   }
});
```

Sombra sobre una parte del mapa

```
CircleOptions circleOptions = new CircleOptions()
        .center(bogota)
        .radius(1000) //metros
        .strokeWidth(10)
        .strokeColor(Color.BLUE)
        .fillColor(Color.argb(128, 127, 0, 0))
        .clickable(true);
mMap.addCircle(circleOptions);
```



https://developers.google.com/maps/documentation/android-api/shapes

Ejercicio 3

- Cree unos limites para búsquedas con Geocoder a direcciones de la región de Bogotá.
- Cree un campo de texto que permita buscar un lugar a partir de una dirección en texto normal. La búsqueda se debe realizar al tocar el botón SEND del teclado virtual.
- Cree un marcador en la localización encontrada del punto anterior y cuya descripción es la dirección encontrada por geocoder.
- Programe su aplicación para que muestre un mapa oscuro en condiciones de baja luminosidad y un mapa claro en condiciones de alta luminosidad.
- Defina un listener que permita crear un marcador tocando el mapa. La descripción del marcador debe ser la dirección encontrada por geocoder.
- Por ultimo, calcule y muestre la distancia entre el punto encontrado por geocoder o creado por el usuario y la localización actual del dispositivo y muéstrelo en un toast.

Mapas

OpenStreetMaps

Otros proveedores

- Open Street Maps: https://github.com/osmdroid/osmdroid
- Here: https://developer.here.com
- Bing: https://www.microsoft.com/en-us/maps/mobile
- Apple maps: https://developer.apple.com/maps/

OpenStreetMaps

- Para usar open street maps, se debe incluir la dependencia en gradle, versión actual: 6.1.11
  - implementation 'org.osmdroid:osmdroid-android:6.1.11’
- Definir el layout de su preferencia e incluir un elemento de tipo MapView:

```
<org.osmdroid.views.MapView
   android:layout_width="match_parent"
   android:layout_height="match_parent"
   android:id="@+id/osmMap">
</org.osmdroid.views.MapView>
```

OpenStreetMaps

- En la actividad, definir un atributo de tipo MapView. En onCreate() conectarlo con el layout (Inflate) y definir el *tilesource*

```
public class OpenStreetMapsActivity extends AppCompatActivity {
   MapView map;
   @Override
   protected void onCreate(Bundle savedInstanceState) {
      super.onCreate(savedInstanceState);
      Context ctx = getApplicationContext();
       Configuration.getInstance().load(ctx,PreferenceManager.getDefaultSharedPreferences(ctx));
      setContentView(R.layout.activity_open_street_maps);
      map =findViewById(R.id.osmMap);
      map.setTileSource(TileSourceFactory.MAPNIK);
      map.setMultiTouchControls(true);
   }
```

OpenStreetMaps

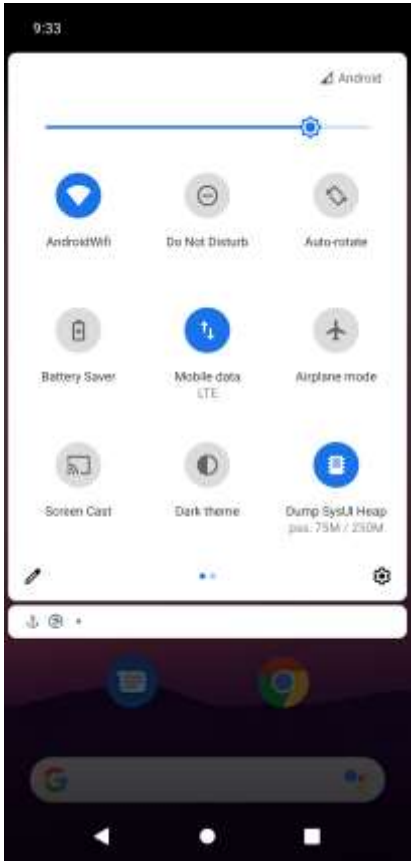
- Sobrecargar los métodos onStart() y onPause() para invocar a los métodos del mapa, se inicializa el mapa con un nivel de 18 en el punto (4.62,-74.07)

```
//starting point for the map
double latitude = 4.62;
double longitude = -74.07;
GeoPoint startPoint = new  GeoPoint(latitude, longitude);
```

```
@Override
protected void onResume() {
    super.onResume();
    map.onResume();
    IMapController mapController = map.getController();
    mapController.setZoom(18.0);
    mapController.setCenter(this.startPoint);
}
@Override
protected void onPause() {
    super.onPause();
    map.onPause();
}
```

Dark mode!

- Desde Android 10, el usuario puede cambiar su interfaz para activar del modo oscuro, el mapa se puede cambiar para reaccionar a este cambio.



Crear el mapa de acuerdo al tema

- En onResume se debe acceder a las propiedades de interfaz para saber si el modo oscuro esta activo. Si lo está, se puede cambiar el estilo del mapa para que se muestre de acuerdo al modo seleccionado por el usuario:

```
@Override
protected void onResume() {
   super.onResume();
   map.onResume();
   IMapController mapController = map.getController();
   mapController.setZoom(18.0);
   mapController.setCenter(this.startPoint);
   UiModeManager uiManager = (UiModeManager) getSystemService(Context.UI_MODE_SERVICE);
   if(uiManager.getNightMode() == UiModeManager.MODE_NIGHT_YES )
      map.getOverlayManager().getTilesOverlay().setColorFilter(TilesOverlay.INVERT_COLORS);
}
```

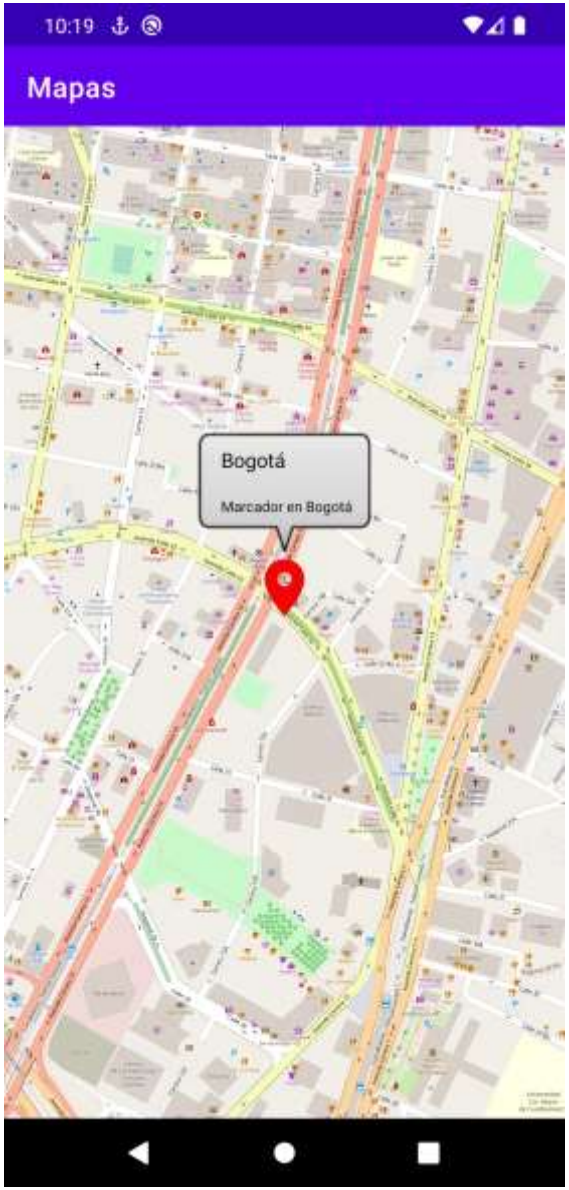
Crear el mapa de acuerdo al tema del usuario

DarkMode OFF

DarkMode ON

Agregar marcadores

```
//Add Marker
GeoPoint markerPoint = new GeoPoint(4.62, -74.07);
Marker marker = new Marker(map);
marker.setTitle("Mi Marcador");
Drawable myIcon =
getResources().getDrawable(R.drawable.location_red,
this.getTheme());
marker.setIcon(myIcon);
marker.setPosition(markerPoint);
marker.setAnchor(Marker.ANCHOR_CENTER,
Marker.ANCHOR_BOTTOM);
map.getOverlays().add(marker);
```



Reaccionar a eventos sobre el mapa

```
//En onCreate
map.getOverlays().add(createOverlayEvents());


private MapEventsOverlay createOverlayEvents(){
   MapEventsOverlay overlayEventos = new MapEventsOverlay(new MapEventsReceiver() {
      @Override
      public boolean singleTapConfirmedHelper(GeoPoint p) {
         return false;
      }
      @Override
      public boolean longPressHelper(GeoPoint p) {
         longPressOnMap(p);
         return true;
      }
   });
   return overlayEventos;
}
```

Reaccionar a eventos sobre el mapa

```
Marker longPressedMarker;

private void longPressOnMap(GeoPoint p){
   if(longPressedMarker!=null)
      map.getOverlays().remove(longPressedMarker);
   longPressedMarker = createMarker(p, "location", null, R.drawable.location_blue);
   map.getOverlays().add(longPressedMarker);
}

private Marker createMarker(GeoPoint p, String title, String desc, int iconID){
   Marker marker = null;
   if(map!=null) {
      marker = new Marker(map);
      if (title != null) marker.setTitle(title);
      if (desc != null) marker.setSubDescription(desc);
      if (iconID != 0) {
         Drawable myIcon = getResources().getDrawable(iconID, this.getTheme());
         marker.setIcon(myIcon);
      }
      marker.setPosition(p);
      marker.setAnchor(Marker.ANCHOR_CENTER, Marker.ANCHOR_BOTTOM);
   }
   return marker;
}
```

Rutas con OSM Bonuspack

- Agregar un nuevo repositorio para dependencias:
  - En graddle.settings:

```
dependencyResolutionManagement {
    repositoriesMode.set(RepositoriesMode.FAIL_ON_PROJECT_REPOS)
    repositories {
        google()
        mavenCentral()
        maven { url 'https://jitpack.io' }
    }
}
```

https://github.com/MKergall/osmbonuspack

Rutas con OSM Bonus Pack

- Agregar la dependencia en el archivo gradle del módulo

```
implementation 'com.github.MKergall:osmbonuspack:6.8.0’
```

- Crear e inicializar un objeto de tipo RoadManager

```
RoadManager roadManager;
//En onCreate
roadManager = new OSRMRoadManager(this, "ANDROID");
```

- Sólo para pruebas, agregar modificar la política de seguridad para permitir llamados síncronos:

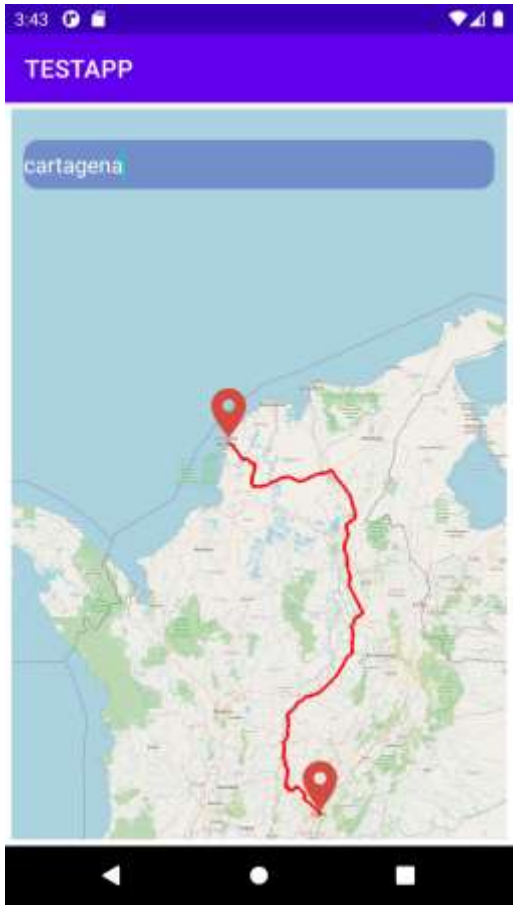
```
//En onCreate
StrictMode.ThreadPolicy policy = new
    StrictMode.ThreadPolicy.Builder().permitAll().build();
StrictMode.setThreadPolicy(policy);
```

https://github.com/MKergall/osmbonuspack

Rutas con OSM Bonuspack

```
//Attribute
Polyline roadOverlay;

private void drawRoute(GeoPoint start, GeoPoint finish){
   ArrayList<GeoPoint> routePoints = new ArrayList<>();
   routePoints.add(start);
   routePoints.add(finish);
   Road road = roadManager.getRoad(routePoints);
   Log.i(IndexActivity.TAG, "Route length: "+road.mLength+" klm");
   Log.i(IndexActivity.TAG, "Duration: "+road.mDuration/60+" min");
   if(map!=null){
      if(roadOverlay!=null){
         map.getOverlays().remove(roadOverlay);
      }
      roadOverlay = RoadManager.buildRoadOverlay(road);
      roadOverlay.getOutlinePaint().setColor(Color.RED);
      roadOverlay.getOutlinePaint().setStrokeWidth(10);
      map.getOverlays().add(roadOverlay);
   }
}
```



https://github.com/MKergall/osmbonuspack

Taller 2!

- Instrucciones en BrightSpace -> General-> Talleres -> Taller 2
- Entrega en parejas durante la sesión 12