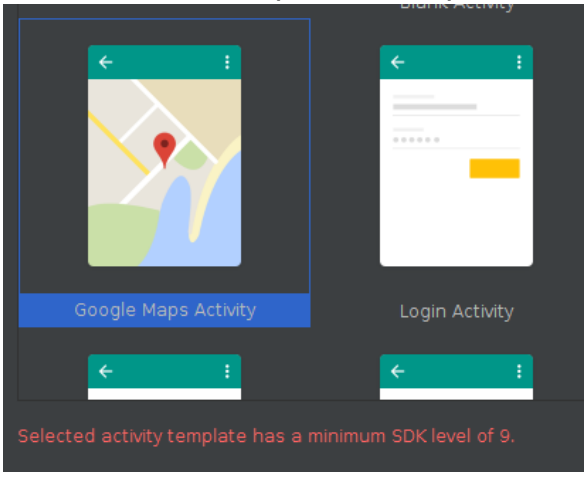
Instituto Tecnológico de Costa Rica

Desarrollo de Aplicaciones Móviles: Laboratorio Google Maps API

Laboratorio Práctico

Paso 1: Creación del Proyecto

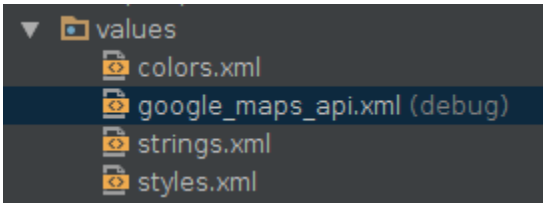
- Cree un proyecto en Android Studio de tipo “Google Maps Activity”
- Puede utilizar cualquier SDK a partir del API 9: Android 2.3 Gingerbread



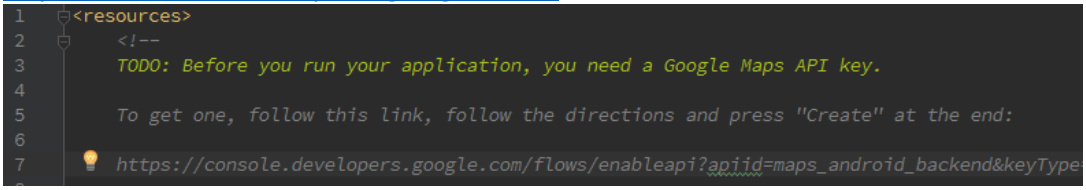
- Se recomienda el uso del API 22: Android 5.1 (Lollipop)

Paso 2: Llave del API de Google Maps

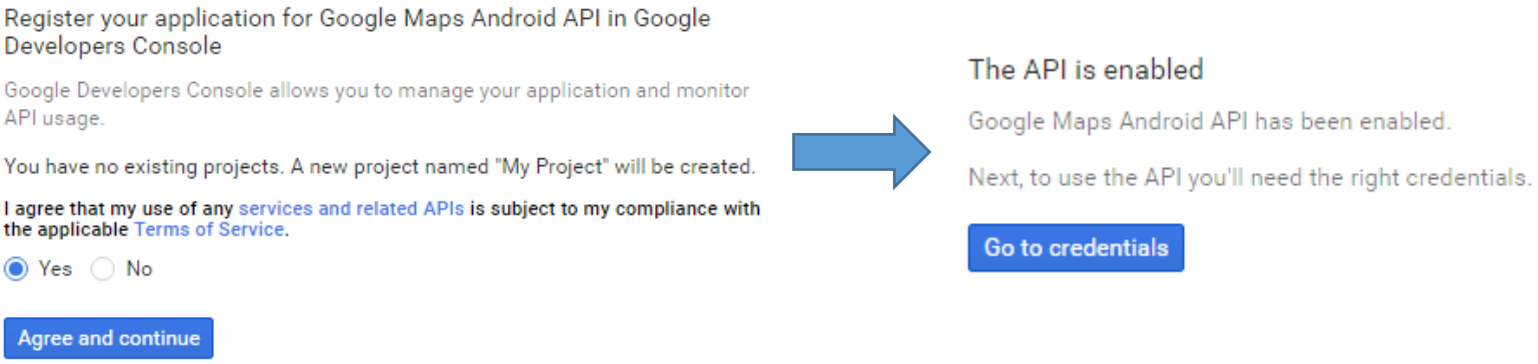
- Para poder acceder al Google Map dentro de su aplicación, deberá de generar una llave de credenciales en la página de desarrolladores de Google.
- Ingrese al archivo google_maps_api.xml dentro de res/values.



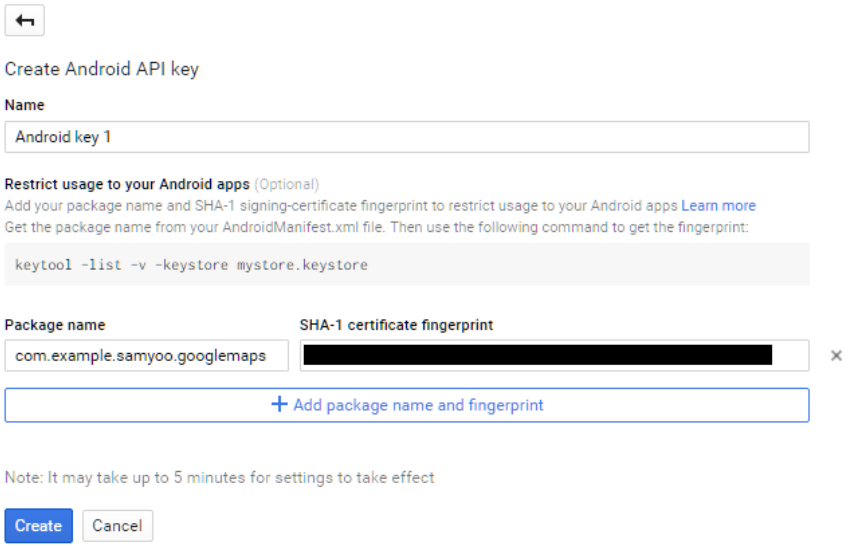
- Ingrese al link señalado en los comentarios del archivo (este es único para su proyecto por lo que debe de copiarlo de ahí). Este comienza con https://console.developers.google.com/...



- Ahí deberá de ingresar con su cuenta de Google, aceptar a la creación de un proyecto nuevo e ingresar a los credenciales.



- Finalmente debe de crear su llave con el certificado SHA-1. Si este no aparece automáticamente, es el código con formato XX:XX:XX:XX:XX:XX proporcionado en los mismos comentarios del archivo google_maps_api.xml



- Le aparecerá un pop-up el cual contendrá la llave que debe de pegar en el archivo google_maps_api.xml


```
<string name="google_maps_key" templateMergeStrategy="preserve" translatable="false">AIza[redacted]</string>
```

Paso 3: Inicio del laboratorio

Instituto Tecnológico de Costa Rica

Desarrollo de Aplicaciones Móviles: Laboratorio Google Maps API

Para el laboratorio, se creará una aplicación simple en donde se mostrarán las principales funcionalidades de GoogleMap.

- Primeramente, reemplace el xml de su actividad principal con el código del archivo provisto llamado activity_maps.xml. Este código simplemente define varios Check Boxes dentro de un Scroll View. Estos serán necesarios para agregarle las funcionalidades después.
- Debe de reemplazar el **extends FragmentActivity** por **extends AppCompatActivity** en la clase principal
- Seguidamente, agregue las siguientes variables y constantes a la clase principal:

```
//Acceso a los distintos ajustes del UI de un GoogleMap
private UiSettings mUiSettings;

//Solo se crean variables para los Check Boxes referentes a la
localización,
// ya que estas son utilizadas para los permisos también
private CheckBox mMyLocationButtonCheckbox;

private CheckBox mMyLocationLayerCheckbox;

//Constantes y variable utilizadasrequeridas para permisos de
localización
private static final int MY_LOCATION_PERMISSION_REQUEST_CODE = 1;

private static final int LOCATION_LAYER_PERMISSION_REQUEST_CODE = 2;

private boolean mLocationPermissionDenied = false;
```

E inicialice los Check Boxes en el onCreate:

```
mMyLocationButtonCheckbox = (CheckBox) findViewById(R.id.mylocationbutton_toggle);
mMyLocationLayerCheckbox = (CheckBox) findViewById(R.id.mylocationlayer_toggle);
```

- Agregue la función isChecked y checkReady. isChecked verifica si un Check Box se encuentra o no marcado (no es necesaria pero es para mayor facilidad), y checkReady verifica que el mapa se encuentra listo para usarse.

```
private boolean isChecked(int id) {
        return ((CheckBox) findViewById(id)).isChecked();
}

/**
 * Verifica que el mapa se encuentre listo.
 * Este método se debe de llamar antes de cualquier método en el GoogleMap
```

- Debe de reemplazar el contenido de onMapReady (esta función se crea por defecto en las actividades de Google Maps) por las siguientes líneas:

```
mMap = googleMap;

mUiSettings = mMap.getUiSettings();

// Habilita todas las funcionalidades al iniciar la aplicación (menos las de
localización, las cuales requieren el permiso del usuario)
mUiSettings.setZoomControlsEnabled(isChecked(R.id.zoom_buttons_toggle));
mUiSettings.setCompassEnabled(isChecked(R.id.compass_toggle));
mUiSettings.setMyLocationButtonEnabled(isChecked(R.id.mylocationbutton_toggle));
mMap.setMyLocationEnabled(isChecked(R.id.mylocationlayer_toggle));
mUiSettings.setScrollGesturesEnabled(isChecked(R.id.scroll_toggle));
mUiSettings.setZoomGesturesEnabled(isChecked(R.id.zoom_gestures_toggle));
mUiSettings.setTiltGesturesEnabled(isChecked(R.id.tilt_toggle));
mUiSettings.setRotateGesturesEnabled(isChecked(R.id.rotate_toggle));
```

Esto obtiene los ajustes de interfaz del mapa y habilita las funcionalidades del mapa (al iniciar el mapa).

- Ahora debemos de crear una función para cada una de las siguientes funcionalidades que se le agregarán al mapa:
  - Botones de zoom
  - Compás
  - Botón/punto de localización
  - Movimiento a través del mapa
  - Gestos para zoom
  - Inclinación
  - Rotación
- Cada una de las funciones es prácticamente igual, solamente debemos de ver el estado del Check Box y con este habilitar/deshabilitar la funcionalidad.
  - Botones de zoom

```
public void setZoomButtonsEnabled(View v) {
      if (!checkReady()) {
            return;
      }
      // Habilita/deshabilita los controles de zoom (Botones +/-)
      mUiSettings.setZoomControlsEnabled(((CheckBox) v).isChecked());
}
```

- Compás

```
public void setCompassEnabled(View v) {
      if (!checkReady()) {
            return;
      }
      // Habilita/deshabilita la funcionalidad del compás (Cuando el mapa se gira,
      // aparece un compás en la parte superior del mapa)
      mUiSettings.setCompassEnabled(((CheckBox) v).isChecked());
}
```

- Botón de localización **(Estos dos son los únicos a los que se le debe de agregar un fragmento de código extra, por los permisos de localización)**

```
public void setMyLocationButtonEnabled(View v) {
      if (!checkReady()) {
            return;
      }
      // Habilita/deshabilita el boton de localizacíon en la parte
      // superior derecha del mapa (esto no habilita o no
      // el punto de localización en el mapa). Para que el
      // botón aparezca, la capa de localización también se debe de activar
      mUiSettings.setMyLocationButtonEnabled(mMyLocationButtonCheckbox.isChecked());

      // Verificación de que se tiene el permiso de localización.
      if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
                  == PackageManager.PERMISSION_GRANTED) {
            mUiSettings.setMyLocationButtonEnabled(mMyLocationButtonCheckbox.isChecked());
      } else {
            // Uncheck the box and request missing location permission.
            mMyLocationButtonCheckbox.setChecked(false);
            requestLocationPermission(MY_LOCATION_PERMISSION_REQUEST_CODE);
      }
}
```

- Punto de localización **(Estos dos son los únicos a los que se le debe de agregar un fragmento de código extra, por los permisos de localización)**

```
public void setMyLocationLayerEnabled(View v) {
      if (!checkReady()) {
            return;
      }
      // Habilita/deshabilita la capa de localización (punto de localización en el mapa).
      // El botón de localización
      // requiere de que esta se encuentre activada para aparecer.

      if (ContextCompat.checkSelfPermission(this, Manifest.permission.ACCESS_FINE_LOCATION)
                  == PackageManager.PERMISSION_GRANTED) {
            mMap.setMyLocationEnabled(mMyLocationLayerCheckbox.isChecked());
```

- Movimiento a través del mapa

```
public void setScrollGesturesEnabled(View v) {
      if (!checkReady()) {
            return;
      }
      // Habilita/deshabilita la función de movilidad a través del mapa.
      mUiSettings.setScrollGesturesEnabled(((CheckBox) v).isChecked());
}
```

- Gestos para zoom

```
public void setZoomGesturesEnabled(View v) {
      if (!checkReady()) {
            return;
      }
      // Habilita/deshabilita la funcionalidad de zoom por medio
      // de gestos (doble tap o con dos dedos)
      mUiSettings.setZoomGesturesEnabled(((CheckBox) v).isChecked());
}
```

- Inclinación

```
public void setTiltGesturesEnabled(View v) {
      if (!checkReady()) {
            return;
      }
      // Habilita/deshabilita la funcionalidad de inclinación del mapa
      // (moviendo dos dedos de forma vertical hacia adelante o atrás)
      mUiSettings.setTiltGesturesEnabled(((CheckBox) v).isChecked());
}
```

```
      if (!checkReady()) {
            return;
      }
      // Habilita/deshabilita los gestos de rotación del mapa ()
      mUiSettings.setRotateGesturesEnabled(((CheckBox) v).isChecked());
}
```

- Con las funciones anteriores, casi todas las funcionalidades del mapa están completas. Lo único faltante es referente a los permisos de localización. Se debe de agregar lo siguiente:

1. Las siguientes 3 funciones al final de la clase principal:

```
public void requestLocationPermission(int requestCode) {
      if (ActivityCompat.shouldShowRequestPermissionRationale(this,
                  Manifest.permission.ACCESS_FINE_LOCATION)) {
            PermissionUtils.RationaleDialog
                        .newInstance(requestCode, false).show(
                        getSupportFragmentManager(), "dialog");
      } else {
            PermissionUtils.requestPermission(this, requestCode,
                        Manifest.permission.ACCESS_FINE_LOCATION, false);
      }
}

@Override
public void onRequestPermissionsResult(int requestCode, @NonNull String[] permissions,
                                                      @NonNull int[] grantResults) {
      if (requestCode == MY_LOCATION_PERMISSION_REQUEST_CODE) {
            if (PermissionUtils.isPermissionGranted(permissions, grantResults,
                        Manifest.permission.ACCESS_FINE_LOCATION)) {
                  mUiSettings.setMyLocationButtonEnabled(true);
                  mMyLocationButtonCheckbox.setChecked(true);
            } else {
                  mLocationPermissionDenied = true;
            }

      } else if (requestCode == LOCATION_LAYER_PERMISSION_REQUEST_CODE) {
            if (PermissionUtils.isPermissionGranted(permissions, grantResults,
                        Manifest.permission.ACCESS_FINE_LOCATION)) {
                  if (ActivityCompat.checkSelfPermission(this,
Manifest.permission.ACCESS_FINE_LOCATION) != PackageManager.PERMISSION_GRANTED &&
ActivityCompat.checkSelfPermission(this, Manifest.permission.ACCESS_COARSE_LOCATION) !=
PackageManager.PERMISSION_GRANTED) {
                        return;
                  }
                  mMap.setMyLocationEnabled(true);
      }
}
```

```
@Override
protected void onResumeFragments() {
      super.onResumeFragments();
      if (mLocationPermissionDenied) {
            PermissionUtils.PermissionDeniedDialog

      .newInstance(false).show(getSupportFragmentManager(), "dialog");
            mLocationPermissionDenied = false;
      }
}
```

2. Se debe de crear una clase Java nueva: PermissionUtils. Esta es provista por Google y se utilizan varios de sus métodos para verificar los permisos de localización. La clase PermissionUtils.java viene incluida en el zip del laboratorio.

Imports finales:

```
import com.google.android.gms.maps.GoogleMap;
import com.google.android.gms.maps.OnMapReadyCallback;
import com.google.android.gms.maps.SupportMapFragment;
import com.google.android.gms.maps.UiSettings;

import android.Manifest;
import android.content.pm.PackageManager;
import android.os.Bundle;
import android.support.annotation.NonNull;
import android.support.v4.app.ActivityCompat;
import android.support.v4.content.ContextCompat;
import android.support.v7.app.AppCompatActivity;
import android.view.View;
import android.widget.CheckBox;
import android.widget.Toast;
```

Preguntas Teóricas:

1. ¿Por qué es necesario incluir una llave de credenciales para el uso de Google Maps (o APIs de Google en general)?
2. Investigue ¿Cuáles son los tipos de mapas que brinda Google Maps?