ArcGIS sdk para QT

Resumen de clases

Mostrar un mapa
Ejemplo en el archivo: A_crearUnMapaBase

Ejemplo 1

Mostrar un mapa

main.cpp

```
const QString apiKey = QString("");
```

Pegue la clave API, adquirida desde su panel de control, entre las comillas

Si corres la app podrás ver tu mapa base

Mostrar un mapa

Archivo.h

Configura tu mapa base para que se pueda observar en un lugar determinado del planeta

```
void setupViewpoint();
```

Agregue la declaración void setupViewpoint() (o como quiera llamar al método); en la parte privada de la clase.

Mostrar un mapa

Archivo.cpp

Configura tu mapa base para que se pueda observar en un lugar determinado del planeta

```
#include "Point.h"
#include "Viewpoint.h"
#include "SpatialReference.h"
#include <QFuture>
```

Agregue las siguientes declaraciones #include.

Mostrar un mapa

Archivo.cpp

Configura tu mapa base para que se pueda observar en un lugar determinado del planeta

```
MapQuickView* Display_a_map::mapView() const { return m_mapView; }
```

```
void Display_a_map::setupViewpoint() {
const Point center(-118.80543, 34.02700, SpatialReference::wgs84());
const Viewpoint viewpoint(center, 100000.0);
m_mapView->setViewpointAsync(viewpoint);
}
```

Agregue código para implementar el método setupViewpoint (o como haya decido llamar a este método).

Este método crea un punto central basado en una referencia espacial junto con la longitud y la latitud. También crea un punto de vista basado en el centro y establece la escala.

Mostrar un mapa

Archivo.cpp


```
setupViewpoint();
```

Agregue la siguiente línea de código para llamar al método setupViewpoint. Esta línea se agrega en dentro método `void Display_a_map::setMapView(MapQuickView* mapView)`

Dibujar líneas, puntos y polígonos
Ejemplo en el archivo: B_dibujarVectores

Ejemplo 2

Dibujar líneas, puntos y polígonos

- *Repetir los pasos del ejercicio anterior (ejemplo 1)*

Dibujar líneas, puntos y polígonos

Archivo.h

```
namespace Esri::ArcGISRuntime {
class Map;
class MapQuickView;
class GraphicsOverlay;
}
```

Agregue la clase GraphicsOverlay a la declaración de ArcGISRuntime.

```
void createGraphics(Esri::ArcGISRuntime::GraphicsOverlay* overlay);
```

Agregue la declaración de función miembro privada (parte privada de la clase) createGraphics() (o como quiera llamarla).

Dibujar líneas, puntos y polígonos

Archivo.cpp

```
#include "Display_a_map.h"
#include "Map.h"
#include "MapQuickView.h"
#include "MapTypes.h"
#include "Point.h"
#include "SpatialReference.h"
#include <QFuture>
#include "Viewpoint.h"
#include "Graphic.h"
#include "GraphicListModel.h"
#include "GraphicsOverlay.h"
#include "GraphicsOverlayListModel.h"
#include "PolylineBuilder.h"
#include "PolygonBuilder.h"
#include "SimpleFillSymbol.h"
#include "SimpleLineSymbol.h"
#include "SimpleMarkerSymbol.h"
#include "SymbolTypes.h"
```

Agregue las siguientes declaraciones #include.

Dibujar líneas, puntos y polígonos

Archivo.cpp

```
void Display_a_map::createGraphics(GraphicsOverlay *overlay) { }
```

Cree un el método que declaro en la parte privada de la clase createGraphics (o como haya decidido llamarlo), justo después del método setupViewpoint (el método que establece la vista del mapa base, recuerde que usted puede darle cualquier nombre a este método).

Dibujar líneas, puntos y polígonos

Archivo.cpp

```
void Display_a_map::createGraphics(GraphicsOverlay *overlay) {

const Point punto(-118.80657463861, 34.0005930608889, SpatialReference::wgs84());
SimpleLineSymbol* contornoPunto= new SimpleLineSymbol(SimpleLineSymbolStyle::DashDot,QColor("red"),1,this);
SimpleMarkerSymbol* simboloPunto = new SimpleMarkerSymbol(SimpleMarkerSymbolStyle::Triangle,QColor("blue"),5, this);
simboloPunto -> setOutline(contornoPunto);
Graphic* graficarPunto = new Graphic(punto,simboloPunto, this);
overlay ->graphics()->append(graficarPunto);

}
```

Agregar un gráfico de puntos

Dibujar líneas, puntos y polígonos

Archivo.cpp

```
void Display_a_map::createGraphics(GraphicsOverlay *overlay) {

const Point punto(-118.80657463861, 34.0005930608889, SpatialReference::wgs84());
SimpleLineSymbol* contornoPunto= new SimpleLineSymbol(SimpleLineSymbolStyle::DashDot,QColor("red"),1,this);
SimpleMarkerSymbol* simboloPunto = new SimpleMarkerSymbol(SimpleMarkerSymbolStyle::Triangle,QColor("blue"),5, this);
simboloPunto -> setOutline(contornoPunto);
Graphic* graficarPunto = new Graphic(punto,simboloPunto, this);
overlay ->graphics()->append(graficarPunto);

PolylineBuilder* polyline_builder = new PolylineBuilder(SpatialReference::wgs84(), this);
polyline_builder->addPoint(-118.8215, 34.0140);
polyline_builder->addPoint(-118.8149, 34.0081);
polyline_builder->addPoint(-118.8089, 34.0017);
SimpleLineSymbol* line_symbol = new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this);
Graphic* polyline_graphic = new Graphic(polyline_builder->toGeometry(), line_symbol, this);
overlay->graphics()->append(polyline_graphic);

}
```

Agregar un gráfico de líneas

Dibujar líneas, puntos y polígonos

Archivo.cpp

```
void Display_a_map::createGraphics(GraphicsOverlay *overlay) {

const Point punto(-118.80657463861, 34.0005930608889, SpatialReference::wgs84());
SimpleLineSymbol* contornoPunto= new SimpleLineSymbol(SimpleLineSymbolStyle::DashDot,QColor("red"),1,this);
SimpleMarkerSymbol* simboloPunto = new SimpleMarkerSymbol(SimpleMarkerSymbolStyle::Triangle,QColor("blue"),5, this);
simboloPunto -> setOutline(contornoPunto);
Graphic* graficarPunto = new Graphic(punto,simboloPunto, this);
overlay ->graphics()->append(graficarPunto);

PolylineBuilder* polyline_builder = new PolylineBuilder(SpatialReference::wgs84(), this);
polyline_builder->addPoint(-118.8215, 34.0140);
polyline_builder->addPoint(-118.8149, 34.0081);
polyline_builder->addPoint(-118.8089, 34.0017);
SimpleLineSymbol* line_symbol = new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this);
Graphic* polyline_graphic = new Graphic(polyline_builder->toGeometry(), line_symbol, this);
overlay->graphics()->append(polyline_graphic);

const QList<Point> points = { Point(-118.8190, 34.0138), Point(-118.8068, 34.0216), Point(-118.7914, 34.0164), Point(-118.7960, 34.0086),
Point(-118.8086, 34.0035), };
PolygonBuilder* polygon_builder = new PolygonBuilder(SpatialReference::wgs84(), this);
polygon_builder->addPoints(points);
SimpleLineSymbol* polygon_line_symbol = new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this);
SimpleFillSymbol* fill_symbol = new SimpleFillSymbol(SimpleFillSymbolStyle::Solid, QColor(Qt::yellow), polygon_line_symbol, this);
Graphic* polygon_graphic = new Graphic(polygon_builder->toGeometry(), fill_symbol, this);
overlay->graphics()->append(polygon_graphic);

}
```

Agregar un gráfico de polígonos

Dibujar líneas, puntos y polígonos

Archivo.cpp

```
GraphicsOverlay* overlay = new GraphicsOverlay(this);
createGraphics(overlay);
m_mapView->graphicsOverlays()->append(overlay);
```

En el método setMapView(), agregue tres líneas de código para crear una GraphicsOverlay, llame al método createGraphics (o como haya decidido llamarlo) y agregue la superposición a la vista del mapa.

Cargar archivos .shp
Ejemplo en el archivo: C_CargarVectores

Ejemplo 3

Cargar archivos .shp

- *Repetir los pasos del ejercicio 1(ejemplo 1). En este caso vamos a cambiar el nombre de la función* `void setupViewpoint();` *por* *void establecerVista();*

Cargar archivos .shp

Archivo.h

```
void CargarCapas();
```

Agregue la declaración void CargarCapas() (o como quiera llamar al método); en la parte privada de la clase.

Dibujar líneas, puntos y polígonos

Archivo.cpp

```
#include "Display_a_map.h"
#include "Map.h"
#include "MapQuickView.h"
#include "MapTypes.h"
#include "Point.h"
#include "SpatialReference.h"
#include <QFuture>
#include "Viewpoint.h"
#include "ShapefileFeatureTable.h"
#include "FeatureLayer.h"
#include "LayerListModel.h"
#include "SimpleRenderer.h"
#include "SimpleMarkerSymbol.h"
#include "SimpleLineSymbol.h"
#include "SimpleFillSymbol.h"
#include "SymbolTypes.h"
```

Agregue las siguientes declaraciones #include.

Cargar archivos .shp

Archivo.cpp

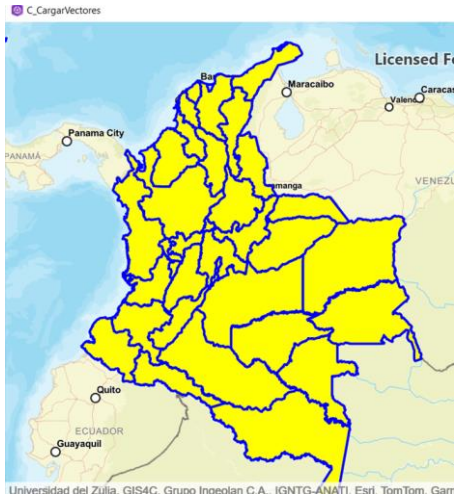
```
void C_CargarVectores::setMapView(MapQuickView
*mapView)
{
   if (!mapView || mapView == m_mapView) {
      return;
   }

   m_mapView = mapView;
   m_mapView->setMap(m_map);
   establecerVista();
   CargarCapas();

   emit mapViewChanged();
}
```

Agregue las siguientes líneas de código para llamar al método establecerVista() y CargarCapas(). Esta línea se agrega en dentro método `void Display_a_map::setMapView(MapQuickView* mapView)`

Cargar archivos .shp

Archivo.cpp

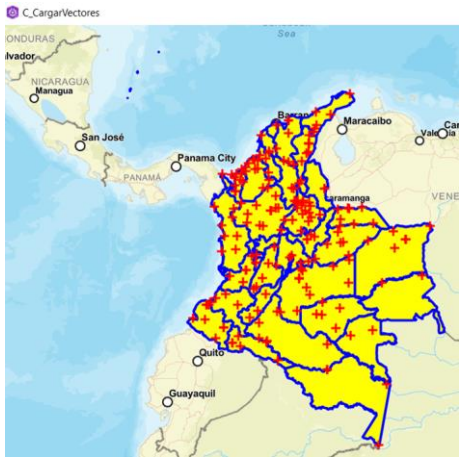
```
void C_CargarVectores::CargarCapas()
{

   QString shapefilePath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/gadm41_COL_1.shp";
   ShapefileFeatureTable* shapefileFeatureTable  = new ShapefileFeatureTable(shapefilePath, this);
   FeatureLayer* featureLayer = new FeatureLayer(shapefileFeatureTable, this);
   m_map->operationalLayers()->append(featureLayer);

   //SimpleRenderer* renderer = new SimpleRenderer(new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this));
   //featureLayer->setRenderer(renderer); //Delineado sin relleno

   SimpleLineSymbol* bordes = new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this);
   SimpleRenderer* renderer = new SimpleRenderer(new SimpleFillSymbol(SimpleFillSymbolStyle::Solid, QColor(Qt::yellow), bordes, this));
   featureLayer->setRenderer(renderer); //Delineado con relleno

}
```



Cargar archivos .shp

Archivo.h

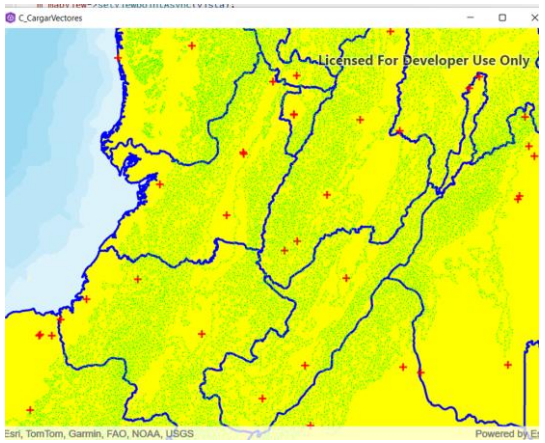
```
void C_CargarVectores::CargarCapas()
{

   QString shapefilePath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/gadm41_COL_1.shp";
   ShapefileFeatureTable* shapefileFeatureTable  = new ShapefileFeatureTable(shapefilePath, this);
   FeatureLayer* featureLayer = new FeatureLayer(shapefileFeatureTable, this);
   m_map->operationalLayers()->append(featureLayer);
   //SimpleRenderer* renderer = new SimpleRenderer(new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this));
   //featureLayer->setRenderer(renderer);
   SimpleLineSymbol* bordes = new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this);
   SimpleRenderer* renderer = new SimpleRenderer(new SimpleFillSymbol(SimpleFillSymbolStyle::Solid, QColor(Qt::yellow), bordes, this));
   featureLayer->setRenderer(renderer);


   QString aeropuertoPath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/Aeropuerto_P.shp";
   ShapefileFeatureTable* aeropuertoFeatureTable  = new ShapefileFeatureTable(aeropuertoPath, this);
   FeatureLayer* aeropuertoLayer = new FeatureLayer(aeropuertoFeatureTable, this);
   m_map->operationalLayers()->append(aeropuertoLayer);
   SimpleRenderer* aeropuertoRenderer = new SimpleRenderer(new SimpleMarkerSymbol(SimpleMarkerSymbolStyle::Cross, QColor("red"), 10.0,
this));
   aeropuertoLayer->setRenderer(aeropuertoRenderer);

}
```



Cargar archivos .shp

Archivo.cpp

```
void C_CargarVectores::CargarCapas()
{

    QString shapefilePath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/gadm41_COL_1.shp";
    ShapefileFeatureTable* shapefileFeatureTable  = new ShapefileFeatureTable(shapefilePath, this);
    FeatureLayer* featureLayer = new FeatureLayer(shapefileFeatureTable, this);
    m_map->operationalLayers()->append(featureLayer);
    //SimpleRenderer* renderer = new SimpleRenderer(new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this));
    //featureLayer->setRenderer(renderer);
    SimpleLineSymbol* bordes = new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this);
    SimpleRenderer* renderer = new SimpleRenderer(new SimpleFillSymbol(SimpleFillSymbolStyle::Solid, QColor(Qt::yellow), bordes, this));
    featureLayer->setRenderer(renderer);


    QString aeropuertoPath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/Aeropuerto_P.shp";
    ShapefileFeatureTable* aeropuertoFeatureTable  = new ShapefileFeatureTable(aeropuertoPath, this);
    FeatureLayer* aeropuertoLayer = new FeatureLayer(aeropuertoFeatureTable, this);
    m_map->operationalLayers()->append(aeropuertoLayer);
    SimpleRenderer* aeropuertoRenderer = new SimpleRenderer(new SimpleMarkerSymbol(SimpleMarkerSymbolStyle::Cross, QColor("red"), 10.0, this));
    aeropuertoLayer->setRenderer(aeropuertoRenderer);

    QString curvasPath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/Curva_Nivel.shp";
    ShapefileFeatureTable* curvasFeatureTable  = new ShapefileFeatureTable(curvasPath, this);
    FeatureLayer* curvasLayer = new FeatureLayer(curvasFeatureTable, this);
    m_map->operationalLayers()->append(curvasLayer);
    SimpleRenderer* curvasRenderer = new SimpleRenderer(new SimpleLineSymbol(SimpleLineSymbolStyle::DashDotDot, QColor(Qt::green), 1, this));
    curvasLayer->setRenderer(curvasRenderer);

}
```



Asignar etiqueta a los archivos .shp
Ejemplo en el archivo: D_CrearEtiquetas

Ejemplo 4

Cargar archivos .shp

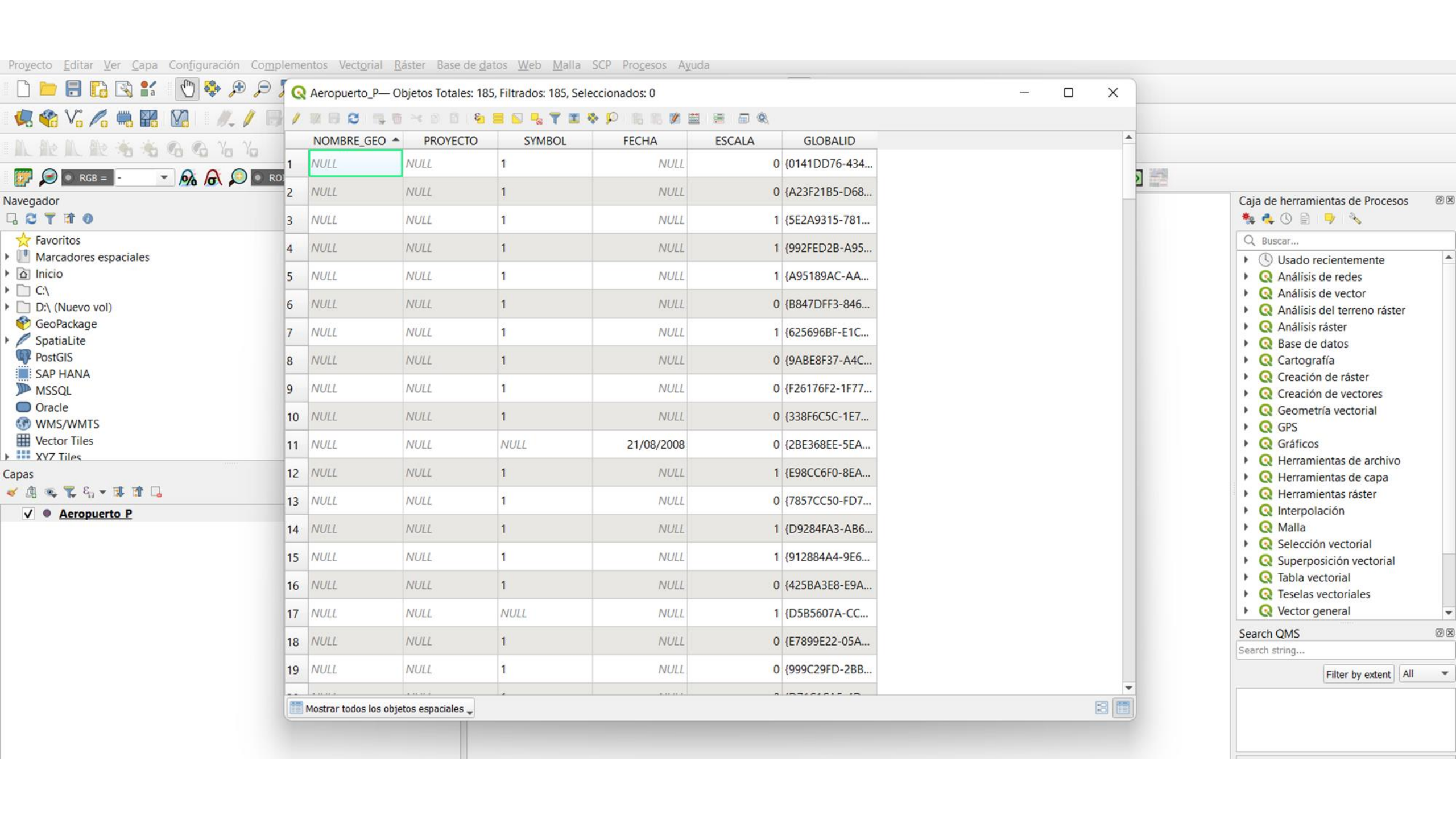
- *Repetir los pasos del ejercicio anterior (ejemplo 3). En este caso vamos a cambiar el nombre de la función* `void setupViewpoint();` *por* *void establecerVista();*

Asignar etiqueta a los archivos .shp

Archivo.cpp

```
#include "Map.h"
#include "MapQuickView.h"
#include "MapTypes.h"

#include "Point.h"
#include "Viewpoint.h"
#include "SpatialReference.h"
#include <QFuture>

#include "ShapefileFeatureTable.h"
#include "FeatureLayer.h"
#include "LayerListModel.h"

#include "SimpleRenderer.h"
#include "SimpleMarkerSymbol.h"
#include "SimpleLineSymbol.h"
#include "SimpleFillSymbol.h"
#include "SymbolTypes.h"

#include "TextSymbol.h"
#include "ArcadeLabelExpression.h"
#include "LabelDefinition.h"
#include "LabelDefinitionListModel.h"
```

Agregue las siguientes declaraciones #include.


Proyecto Editar Ver Capa Configuración Complementos Vectorial Ráster Base de datos Web Malla SCP Procesos Ayuda
Aeropuerto_P— Objetos Totales: 185, Filtrados: 185, Seleccionados: 0
NOMBRE_GEO PROYECTO SYMBOL FECHA ESCALA GLOBALID
1 NULL NULL 1 NULL 0 {0141DD76-434...
2 NULL NULL 1 NULL 0 {A23F21B5-D68...
3 NULL NULL 1 NULL 1 {5E2A9315-781...
4 NULL NULL 1 NULL 1 {992FED2B-A95...
5 NULL NULL 1 NULL 1 {A95189AC-AA...
6 NULL NULL 1 NULL 0 {B847DFF3-846...
7 NULL NULL 1 NULL 1 {625696BF-E1C...
8 NULL NULL 1 NULL 0 {9ABE8F37-A4C...
9 NULL NULL 1 NULL 0 {F26176F2-1F77...
10 NULL NULL 1 NULL 0 {338F6C5C-1E7...
11 NULL NULL NULL 21/08/2008 0 {2BE368EE-5EA...
12 NULL NULL 1 NULL 1 {E98CC6F0-8EA...
13 NULL NULL 1 NULL 0 {7857CC50-FD7...
14 NULL NULL 1 NULL 1 {D9284FA3-AB6...
15 NULL NULL 1 NULL 1 {912884A4-9E6...
16 NULL NULL 1 NULL 0 {425BA3E8-E9A...
17 NULL NULL NULL NULL 1 {D5B5607A-CC...
18 NULL NULL 1 NULL 0 {E7899E22-05A...
19 NULL NULL 1 NULL 0 {999C29FD-2BB...
Mostrar todos los objetos espaciales
Navegador
Favoritos
Marcadores espaciales
Inicio
C:\
D:\ (Nuevo vol)
GeoPackage
SpatiaLite
PostGIS
SAP HANA
MSSQL
Oracle
WMS/WMTS
Vector Tiles
XYZ Tiles
Capas
Aeropuerto P
Caja de herramientas de Procesos
Buscar...
Usado recientemente
Análisis de redes
Análisis de vector
Análisis del terreno ráster
Análisis ráster
Base de datos
Cartografía
Creación de ráster
Creación de vectores
Geometría vectorial
GPS
Gráficos
Herramientas de archivo
Herramientas de capa
Herramientas ráster
Interpolación
Malla
Selección vectorial
Superposición vectorial
Tabla vectorial
Teselas vectoriales
Vector general
Search QMS
Search string...
Filter by extent
All

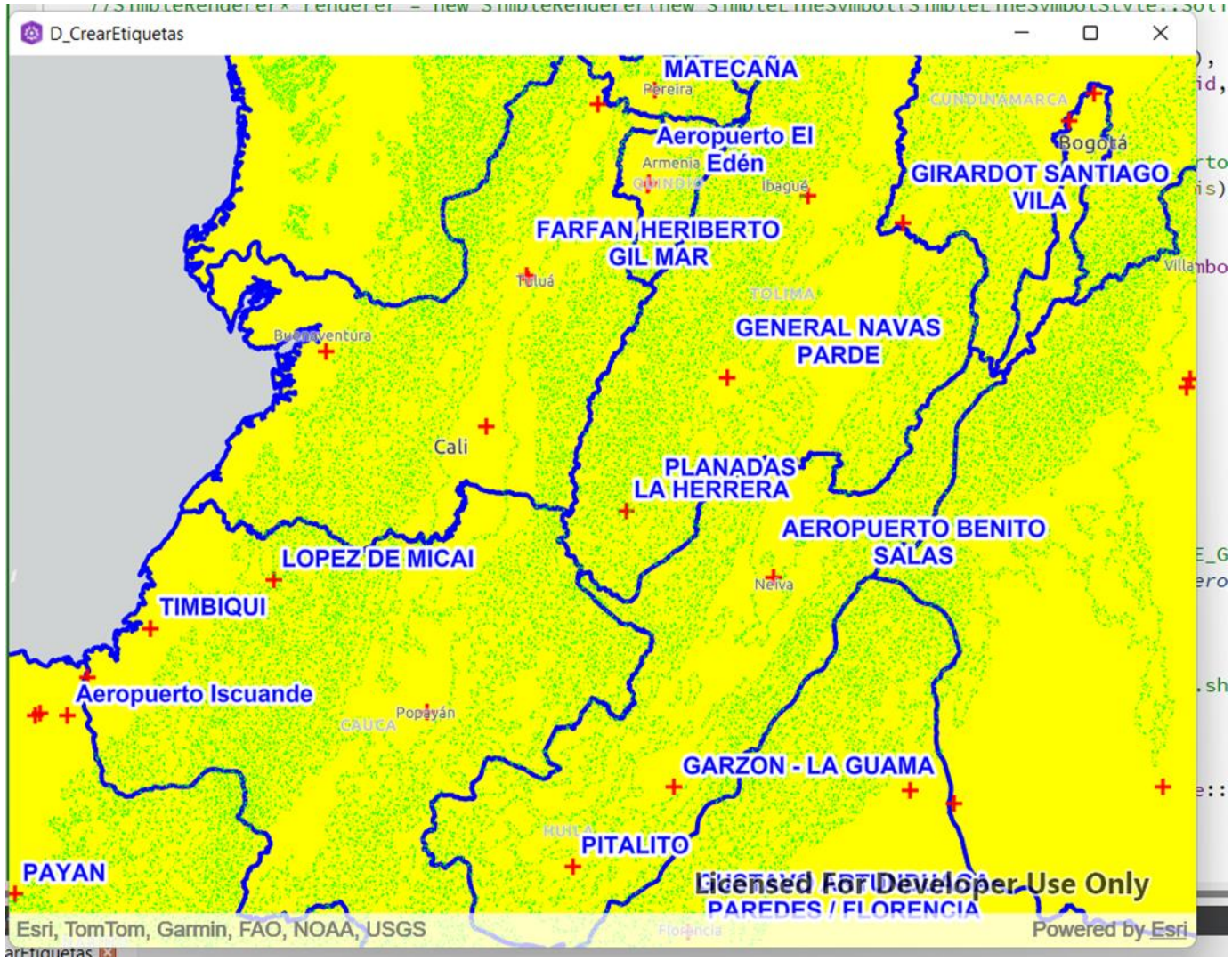
Asignar etiqueta a los archivos .shp

Archivo.cpp

```
void C_CargarVectores::CargarCapas()
{

    QString shapefilePath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/gadm41_COL_1.shp";
    ShapefileFeatureTable* shapefileFeatureTable  = new ShapefileFeatureTable(shapefilePath, this);
    FeatureLayer* featureLayer = new FeatureLayer(shapefileFeatureTable, this);
    m_map->operationalLayers()->append(featureLayer);
    //SimpleRenderer* renderer = new SimpleRenderer(new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this));
    //featureLayer->setRenderer(renderer);
    SimpleLineSymbol* bordes = new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this);
    SimpleRenderer* renderer = new SimpleRenderer(new SimpleFillSymbol(SimpleFillSymbolStyle::Solid, QColor(Qt::yellow), bordes, this));
    featureLayer->setRenderer(renderer);

    QString aeropuertoPath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/Aeropuerto_P.shp";
    ShapefileFeatureTable* aeropuertoFeatureTable  = new ShapefileFeatureTable(aeropuertoPath, this);
    FeatureLayer* aeropuertoLayer = new FeatureLayer(aeropuertoFeatureTable, this);
    m_map->operationalLayers()->append(aeropuertoLayer);
    SimpleRenderer* aeropuertoRenderer = new SimpleRenderer(new SimpleMarkerSymbol(SimpleMarkerSymbolStyle::Cross, QColor("red"), 10.0, this));
    aeropuertoLayer->setRenderer(aeropuertoRenderer);
    aeropuertoLayer->setLabelsEnabled(true);

    TextSymbol* AeropuertosTextSymbol = new TextSymbol(this);
    AeropuertosTextSymbol->setFontFamily("Arial");
    AeropuertosTextSymbol->setFontWeight(FontWeight::Bold);
    AeropuertosTextSymbol->setSize(16);
    AeropuertosTextSymbol->setColor(QColor("blue"));
    AeropuertosTextSymbol->setHaloColor(QColor("white"));
    AeropuertosTextSymbol->setHaloWidth(2);
    ArcadeLabelExpression* AeropuertoLabelExpression  = new ArcadeLabelExpression("$feature.NOMBRE_GEO", this);
    LabelDefinition* AeropuertoLabelDefinition = new LabelDefinition(AeropuertoLabelExpression, AeropuertosTextSymbol, this);
    AeropuertoLabelDefinition->setWhereClause("[SYMBOL] >= 1");
    aeropuertoLayer->labelDefinitions()->append(AeropuertoLabelDefinition);

    QString curvasPath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/Curva_Nivel.shp";
    ShapefileFeatureTable* curvasFeatureTable  = new ShapefileFeatureTable(curvasPath, this);
    FeatureLayer* curvasLayer = new FeatureLayer(curvasFeatureTable, this);
    m_map->operationalLayers()->append(curvasLayer);
    SimpleRenderer* curvasRenderer = new SimpleRenderer(new SimpleLineSymbol(SimpleLineSymbolStyle::DashDotDot, QColor(Qt::green), 1, this));
    curvasLayer->setRenderer(curvasRenderer);

}
```

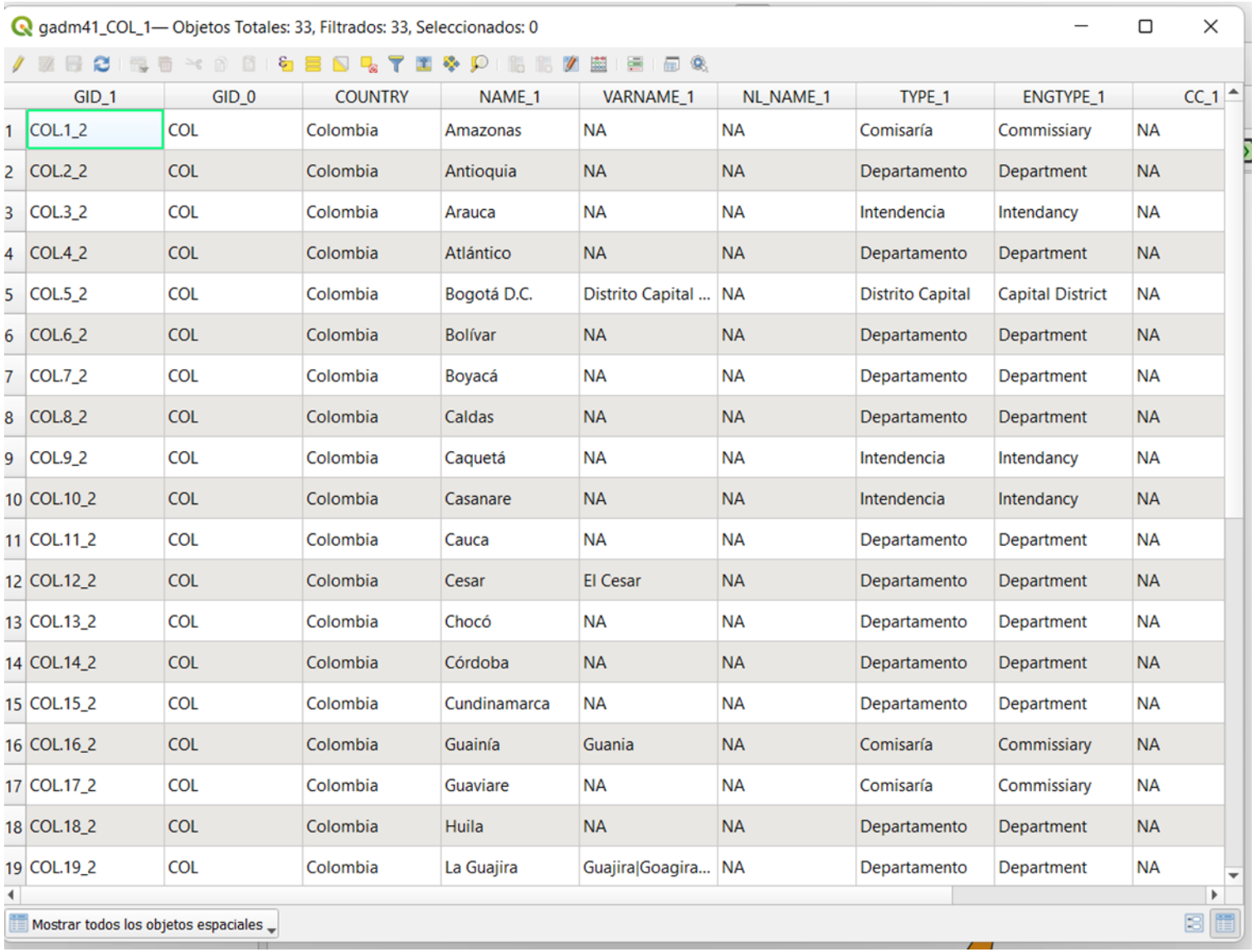
Asignar etiqueta a los archivos .shp

Archivo.cpp



Asignar etiqueta a los archivos .shp

Archivo.cpp



	GID_1	GID_0	COUNTRY	NAME_1	VARNAME_1	NL_NAME_1	TYPE_1	ENGTYPE_1	CC_1
1	COL.1_2	COL	Colombia	Amazonas	NA	NA	Comisaría	Commissiary	NA
2	COL.2_2	COL	Colombia	Antioquia	NA	NA	Departamento	Department	NA
3	COL.3_2	COL	Colombia	Arauca	NA	NA	Intendencia	Intendancy	NA
4	COL.4_2	COL	Colombia	Atlántico	NA	NA	Departamento	Department	NA
5	COL.5_2	COL	Colombia	Bogotá D.C.	Distrito Capital ...	NA	Distrito Capital	Capital District	NA
6	COL.6_2	COL	Colombia	Bolívar	NA	NA	Departamento	Department	NA
7	COL.7_2	COL	Colombia	Boyacá	NA	NA	Departamento	Department	NA
8	COL.8_2	COL	Colombia	Caldas	NA	NA	Departamento	Department	NA
9	COL.9_2	COL	Colombia	Caquetá	NA	NA	Intendencia	Intendancy	NA
10	COL.10_2	COL	Colombia	Casanare	NA	NA	Intendencia	Intendancy	NA
11	COL.11_2	COL	Colombia	Cauca	NA	NA	Departamento	Department	NA
12	COL.12_2	COL	Colombia	Cesar	El Cesar	NA	Departamento	Department	NA
13	COL.13_2	COL	Colombia	Chocó	NA	NA	Departamento	Department	NA
14	COL.14_2	COL	Colombia	Córdoba	NA	NA	Departamento	Department	NA
15	COL.15_2	COL	Colombia	Cundinamarca	NA	NA	Departamento	Department	NA
16	COL.16_2	COL	Colombia	Guainía	Guania	NA	Comisaría	Commissiary	NA
17	COL.17_2	COL	Colombia	Guaviare	NA	NA	Comisaría	Commissiary	NA
18	COL.18_2	COL	Colombia	Huila	NA	NA	Departamento	Department	NA
19	COL.19_2	COL	Colombia	La Guajira	Guajira|Goagira...	NA	Departamento	Department	NA

Mostrar un mapa

Archivo.cpp

```
QString shapefilePath = "D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/shapes/gadm41_COL_1.shp";
    ShapefileFeatureTable* shapefileFeatureTable  = new ShapefileFeatureTable(shapefilePath, this);
    FeatureLayer* featureLayer = new FeatureLayer(shapefileFeatureTable, this);
    m_map->operationalLayers()->append(featureLayer);
    //SimpleRenderer* renderer = new SimpleRenderer(new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this));
    //featureLayer->setRenderer(renderer);
    SimpleLineSymbol* bordes = new SimpleLineSymbol(SimpleLineSymbolStyle::Solid, QColor(Qt::blue), 3, this);
    SimpleRenderer* renderer = new SimpleRenderer(new SimpleFillSymbol(SimpleFillSymbolStyle::Solid, QColor(Qt::yellow), bordes, this));
    featureLayer->setRenderer(renderer);
    featureLayer->setLabelsEnabled(true);

    TextSymbol* PoligonosTextSymbol = new TextSymbol(this);
    PoligonosTextSymbol->setFontFamily("Arial");
    PoligonosTextSymbol->setFontWeight(FontWeight::Bold);
    PoligonosTextSymbol->setSize(16);
    PoligonosTextSymbol->setColor(QColor("blue"));
    PoligonosTextSymbol->setHaloColor(QColor("white"));
    PoligonosTextSymbol->setHaloWidth(2);
    ArcadeLabelExpression* PoligonosLabelExpression  = new ArcadeLabelExpression("$feature.NAME_1", this);
    LabelDefinition* PoligonoLabelDefinition = new LabelDefinition(PoligonosLabelExpression, PoligonosTextSymbol, this);
    featureLayer->labelDefinitions()->append(PoligonoLabelDefinition);
```

Para los Polígonos

Cargar archivos raster
Ejemplo en el archivo: E_ArchivosRaster

Ejemplo 5

Cargar archivos raster

Archivo.h

```
private:
    Esri::ArcGISRuntime::MapQuickView *mapView() const;
    void setMapView(Esri::ArcGISRuntime::MapQuickView *mapView);
    void vista();
    Esri::ArcGISRuntime::Map *m_map = nullptr;
    Esri::ArcGISRuntime::MapQuickView *m_mapView = nullptr;
};
```

Declarar la función de vista

Cargar archivos raster

Archivo.cpp

```
#include "Map.h"
#include "MapQuickView.h"
#include "MapTypes.h"

#include "Raster.h"
#include "RasterLayer.h"
#include "LayerListModel.h"

#include "Basemap.h"
#include "ColormapRenderer.h"
QList<QColor> colormap;
```

Agregue las siguientes declaraciones #include.

Cargar archivos raster

Archivo.cpp

```
void E_ArchivosRaster::vista(){
    Raster* raster = new Raster("D:/DOCS/ASIGNATURAS IMPARTIDAS/ArcGIS SDK/ResumerPro/raster/Alturas de California.tif", this);
    RasterLayer* rasterLayer  = new RasterLayer(raster, this);
    colormap << QColor(Qt::red) << QColor(Qt::lightGray) << QColor(Qt::green)<< QColor(Qt::black)<< QColor(Qt::white)<< QColor(Qt::yellow);
    ColormapRenderer* colormapRenderer = new ColormapRenderer(colormap,  this);
    rasterLayer->setRenderer(colormapRenderer);
    //Basemap* basemap = new Basemap(rasterLayer, this);
    //Map* map = new Map(basemap, this);
    //m_mapView->setMap(map);
    m_map->operationalLayers()->append(rasterLayer);
}
```

Agregue código para implementar el método vista (o como haya decido llamar a este método).

Cargar archivos raster

Archivo.h

```
void E_ArchivosRaster::setMapView(MapQuickView *mapView)
{
   if (!mapView || mapView == m_mapView) {
      return;
   }

   m_mapView = mapView;
   m_mapView->setMap(m_map);
   vista();
   emit mapViewChanged();
}
```

Agregue la siguiente línea de código para llamar al método vista(). Esta línea se agrega en dentro método `void Display_a_map::setMapView(MapQuickView* mapView)`