# Creating map projects in QGIS

Setting parameters of map projects for further publication in eLiteGIS

## Table of contents

# 1. Introduction

## 1.1. CoGIS platform components

CoGIS platform consists of the following software components:

- **CoGIS Designer** – a constructor for creation of interactive maps and fully functional web map applications based on map services, geoprocessing and analyses tools;
- **CoGIS SOE** (SOE, an abbreviation for Server Object Extension) – a module providing support for advanced methods to work with the map services layers and objects;
- **CoGIS Portal –** a geoportal consisting of catalog of published interactive maps and map apps, tools for searching and navigation, and web pages with reference information which structure and content are set in accordance with the users' needs;
- **CoGIS Mobile –** mobile applications for work with interactive maps and map apps on iOS and Android devices and mobile service for operation of these applications;
- **eLiteGIS** – a GIS server for publishing data and tools as web services.

## eLiteGIS components

- Sever components provided for publishing of services and arranging web access to these services via REST API;
- Web console **eLiteGIS Server Manager** with graphic interface for publishing GIS services and setting GIS server.

**eLiteGIS** allows you to publish various types of services, including map services: dynamic and tile, open only for viewing or also for editing; with vector and raster layers.

One source of data for publishing map services can be a map project in QGIS format, created by means of desktop GIS software QGIS.

The given manual provides instructions on setting selected parameters of QGIS projects, allowing you to get advantage of using the extended functionality on working with geodata by publishing this data as services in eLiteGIS.

The complete list of available manuals is provided in section **Ошибка! Источник ссылки не найден.**.

This manual contains instructions on creating and setting of interactive maps and web map applications in CoGIS Designer, and also setting of Maps catalog in CoGIS Portal.

Complete list of instructions on work with platform components is provided in section 1.2 below.

## 1.2. Additional information

The following manuals with information about CoGIS platform can be also helpful:

- Publishing GIS services in eLiteGIS;
- Installing and setting eLiteGIS;
- Installing and setting CoGIS;
- Creating map projects in QGIS;

- Creating map applications in CoGIS;
- Working in mobile applications CoGIS Mobile.

## 2. Settings of connection to database

### 2.1. Supported DBMS

**eLiteGIS** supports the work with map projects, the data sources of which can be MSSQL or PostgreSQL/PostGIS databases.

### 2.2. Connection to database

To connect to database in QGIS, select DBMS as shown on Figure 1.

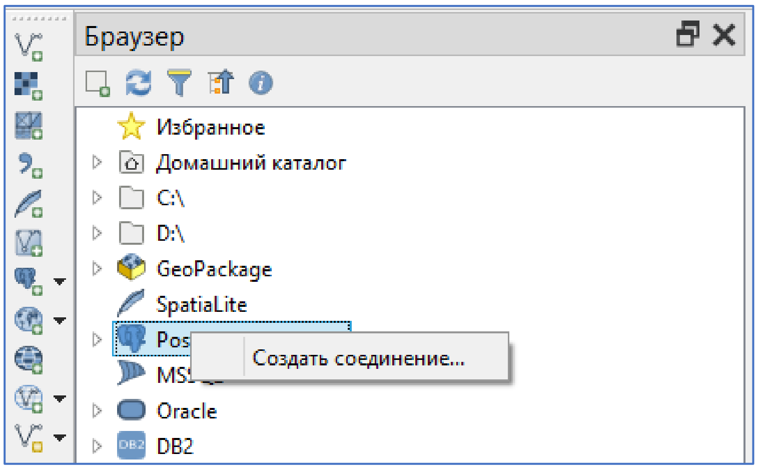
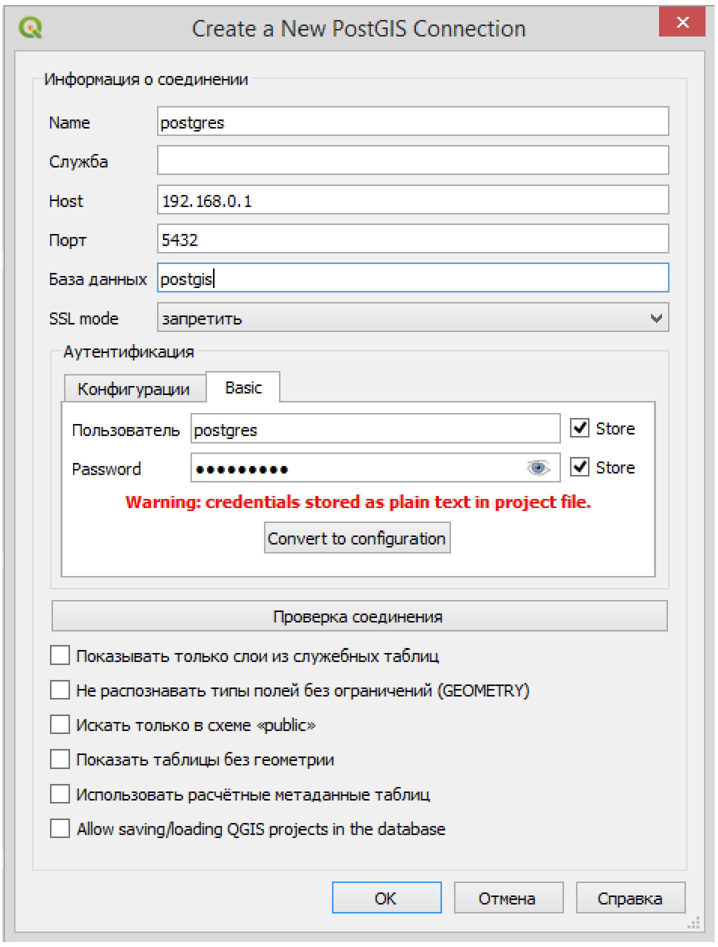


Figure 1 – Connection to database

Now enter the connection parameters (*Host*, *Port*, *Login*, *Password* etc.), see Figure 2.


Create a New PostGIS Connection

Информация о соединении

| | |
|---|---|
| Name | postgres |
| Служба | |
| Host | 192.168.0.1 |
| Порт | 5432 |
| База данных | postgis |
| SSL mode | запретить |

Аутентификация

Конфигурации | Basic

| | | |
|---|---|---|
| Пользователь | postgres | ✓ Store |
| Password | ●●●●●●●●● | ✓ Store |

Warning: credentials stored as plain text in project file.

Convert to configuration

Проверка соединения

- [ ] Показывать только слои из служебных таблиц
- [ ] Не распознавать типы полей без ограничений (GEOMETRY)
- [ ] Искать только в схеме «public»
- [ ] Показать таблицы без геометрии
- [ ] Использовать расчётные метаданные таблиц
- [ ] Allow saving/loading QGIS projects in the database

OK | Отмена | Справка

Figure 2 – Entering connection parameters

## 2.3. Database manager

Working with the connected database in QGIS is recommended via Database manager, see Figure 3.

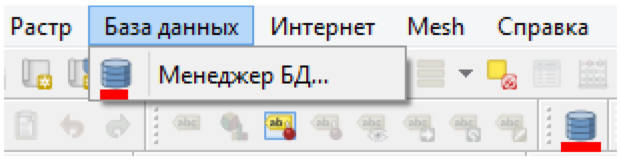


Figure 3 – Database manager in QGIS

### 2.3.1. Creating table

To create the table in the database, select the appropriate menu option in Database manager, see Figure 4.

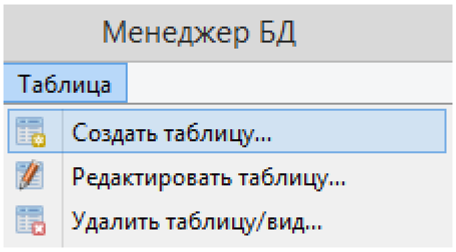
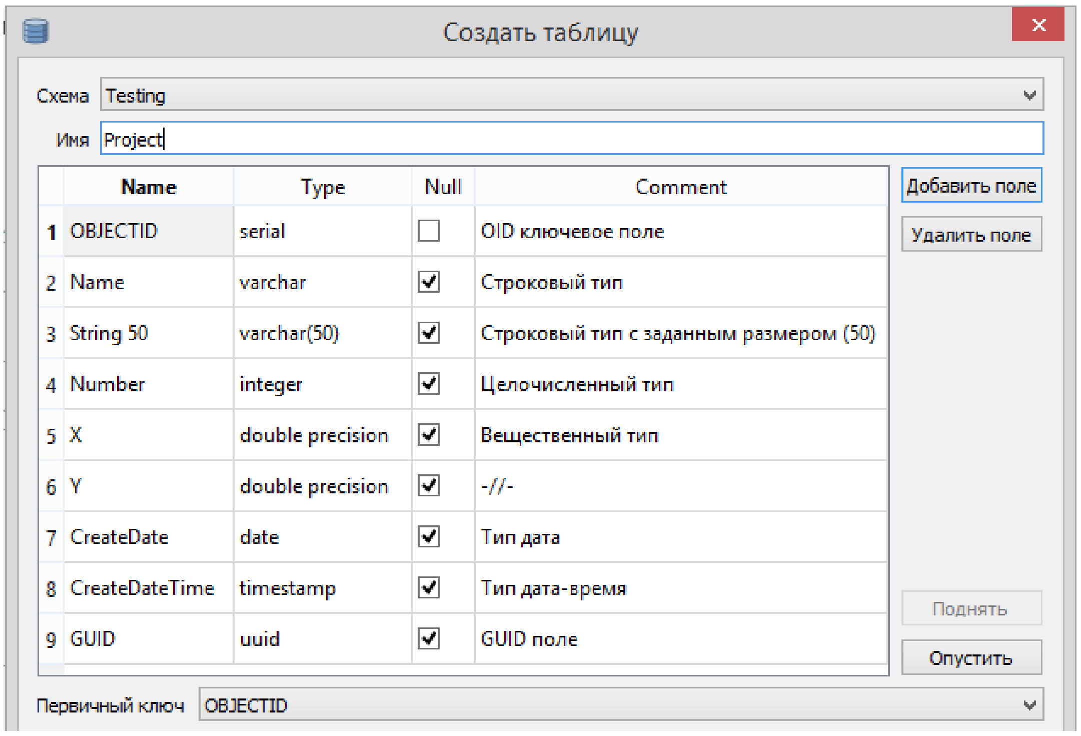


Figure 4 – Creation of new table in database

### 2.3.2. General types of fields in the table

The list of general types of fields in the table is shown on Figure 5.



Создать таблицу

Схема: Testing

Имя: Project

| | Name | Type | Null | Comment |
|---|---|---|---|---|
| 1 | OBJECTID | serial | ☐ | OID ключевое поле |
| 2 | Name | varchar | ✓ | Строковый тип |
| 3 | String 50 | varchar(50) | ✓ | Строковый тип с заданным размером (50) |
| 4 | Number | integer | ✓ | Целочисленный тип |
| 5 | X | double precision | ✓ | Вещественный тип |
| 6 | Y | double precision | ✓ | -//- |
| 7 | CreateDate | date | ✓ | Тип дата |
| 8 | CreateDateTime | timestamp | ✓ | Тип дата-время |
| 9 | GUID | uuid | ✓ | GUID поле |

Первичный ключ: OBJECTID

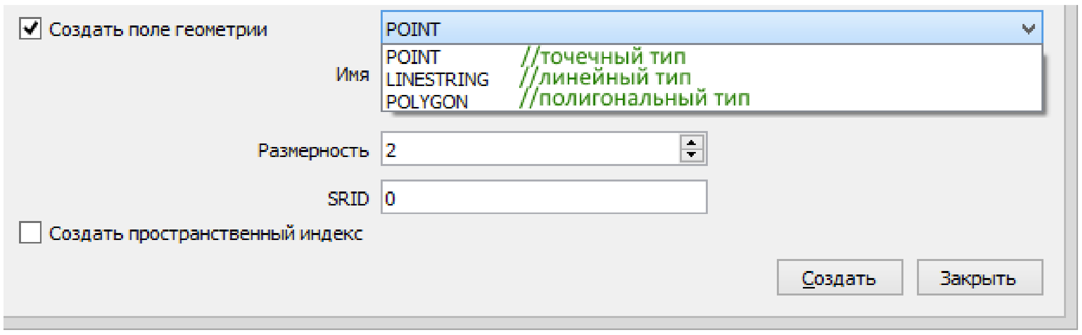
Figure 5 – General types of fields in the table

The checked Null box means that the empty value is allowed.

The *Serial* type is the integer type (*int*) with the autoincrement.

### 2.3.3. General types of geometry fields

The tables in the database support point, polyline and polygon types of geometry. To create the required field type, select the appropriate option as shown on Figure 6.



[✓] Создать поле геометрии

Имя: POINT

```
POINT        //точечный тип
LINESTRING   //линейный тип
POLYGON      //полигональный тип
```

Размерность: 2

SRID: 0

[ ] Создать пространственный индекс

Figure 6 – Creating geometry field

### 2.3.4. Importing data from file or layer

To the database table it is possible to import data from the existing layer in QGIS project or from the file with data, see Figure 7.

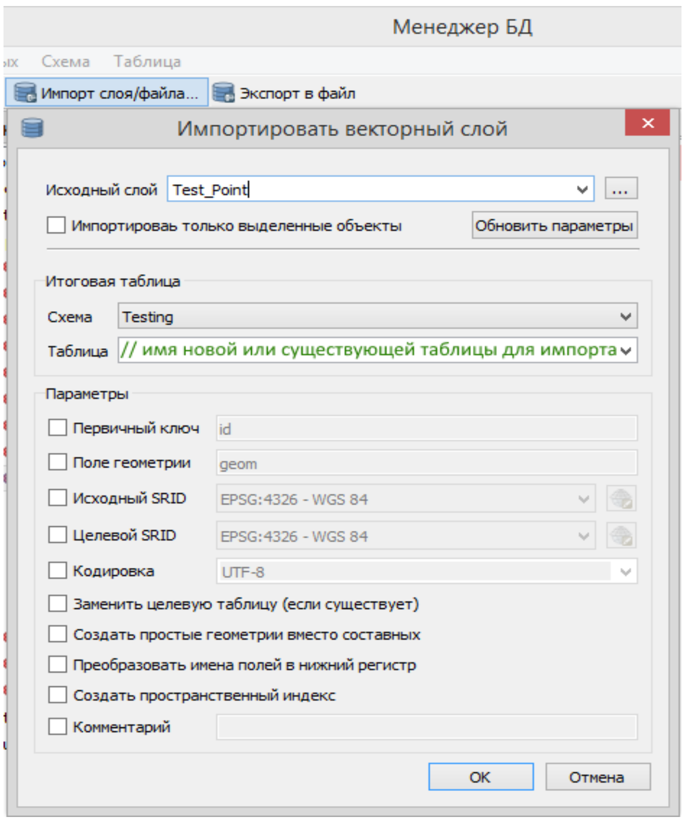

Менеджер БД

Схема Таблица

Импорт слоя/файла... Экспорт в файл

Импортировать векторный слой

Исходный слой: Test_Point

☐ Импортироваь только выделенные объекты — Обновить параметры

Итоговая таблица

Схема: Testing

Таблица: // имя новой или существующей таблицы для импорта

Параметры

| | |
|---|---|
| ☐ Первичный ключ | id |
| ☐ Поле геометрии | geom |
| ☐ Исходный SRID | EPSG:4326 - WGS 84 |
| ☐ Целевой SRID | EPSG:4326 - WGS 84 |
| ☐ Кодировка | UTF-8 |
| ☐ Заменить целевую таблицу (если существует) | |
| ☐ Создать простые геометрии вместо составных | |
| ☐ Преобразовать имена полей в нижний регистр | |
| ☐ Создать пространственный индекс | |
| ☐ Комментарий | |

OK Отмена

Figure 7 – Importing data from layer or file to table

### 2.3.5. Adding table to map of the project

The table with spatial data can be added to the project map as shown on Figure 8.

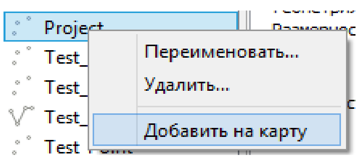
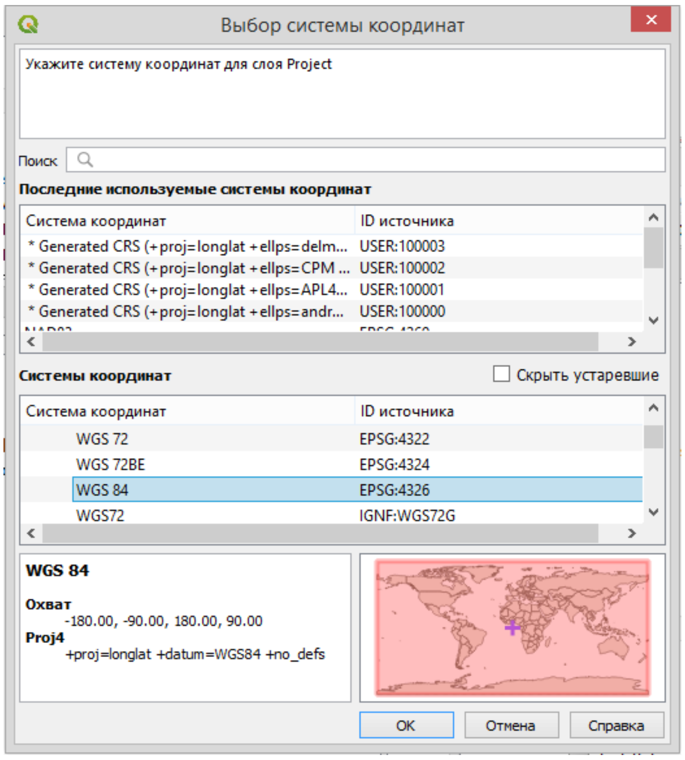


Figure 8 – Adding table to the project map

Now you need to select coordinate system for the added layer, see Figure 9.


Выбор системы координат

Укажите систему координат для слоя Project

Поиск

**Последние используемые системы координат**

| Система координат | ID источника |
|---|---|
| * Generated CRS (+proj=longlat +ellps=delm... | USER:100003 |
| * Generated CRS (+proj=longlat +ellps=CPM ... | USER:100002 |
| * Generated CRS (+proj=longlat +ellps=APL4... | USER:100001 |
| * Generated CRS (+proj=longlat +ellps=andr... | USER:100000 |

**Системы координат** ☐ Скрыть устаревшие

| Система координат | ID источника |
|---|---|
| WGS 72 | EPSG:4322 |
| WGS 72BE | EPSG:4324 |
| WGS 84 | EPSG:4326 |
| WGS72 | IGNF:WGS72G |

**WGS 84**

**Охват**
-180.00, -90.00, 180.00, 90.00

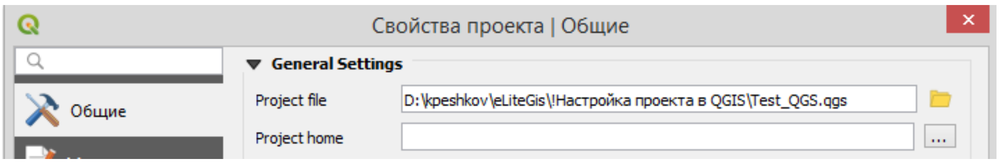
**Proj4**
+proj=longlat +datum=WGS84 +no_defs

Figure 9 – Selecting coordinate system for the added layer

# 3. QGS project's properties

## 3.1. General properties

One of the important properties of the QGS project is the path to the project's file. It is required to specify or to change this path, see Figure 10.



Свойства проекта | Общие

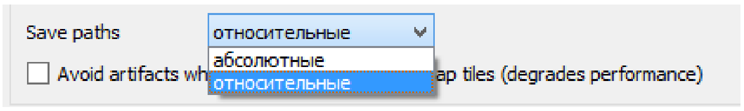
Общие

**General Settings**

| | |
|---|---|
| Project file | D:\kpeshkov\eLiteGis\!Настройка проекта в QGIS\Test_QGS.qgs |
| Project home | |

Figure 10 – General project's properties

Now you need to set the saved paths to the data sources as shown on Figure 11.



Save paths: относительные

абсолютные

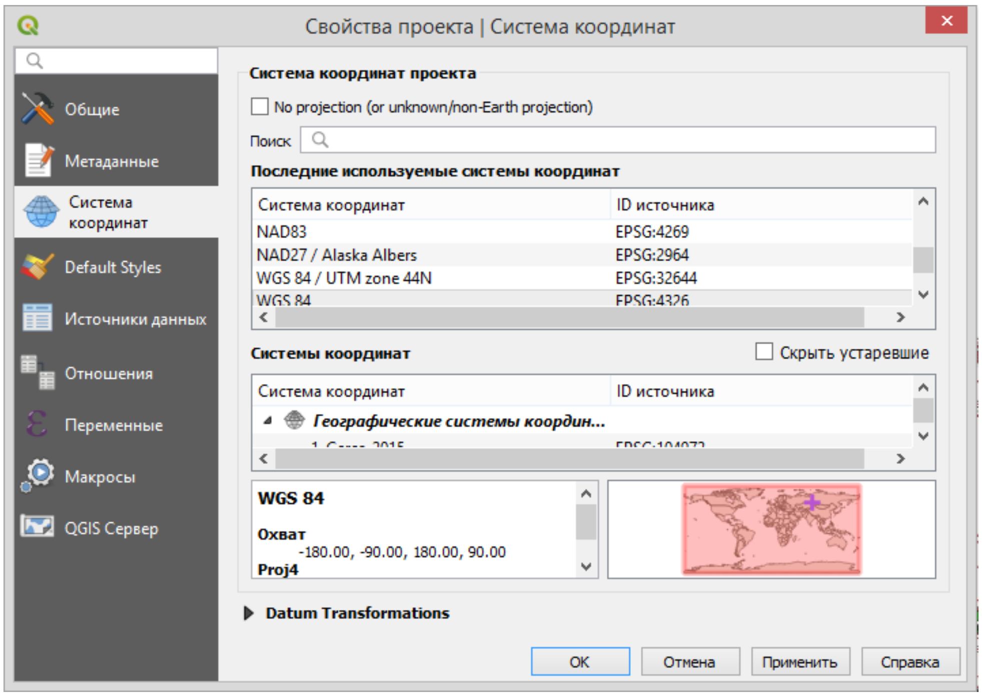
относительные

☐ Avoid artifacts wh… ap tiles (degrades performance)

Figure 11 – Paths to data sources

## 3.2. Coordinate system

For the QGIS project you need to set the coordinate system, see Figure 12.



Свойства проекта | Система координат

**Система координат проекта**

☐ No projection (or unknown/non-Earth projection)

Поиск

**Последние используемые системы координат**

| Система координат | ID источника |
|---|---|
| NAD83 | EPSG:4269 |
| NAD27 / Alaska Albers | EPSG:2964 |
| WGS 84 / UTM zone 44N | EPSG:32644 |
| WGS 84 | EPSG:4326 |

**Системы координат** ☐ Скрыть устаревшие

| Система координат | ID источника |
|---|---|
| ***Географические системы координ…*** | |

**WGS 84**

**Охват**
-180.00, -90.00, 180.00, 90.00

**Proj4**

**Datum Transformations**

Figure 12 – Setting coordinate system for the project

*No projection* option is not supported. The coordinate system needs to be always set for the project.

## 3.3. Default styles

For the project you need to set the default styles (*Default Styles* -> *Managing styles*), see Figure 13.

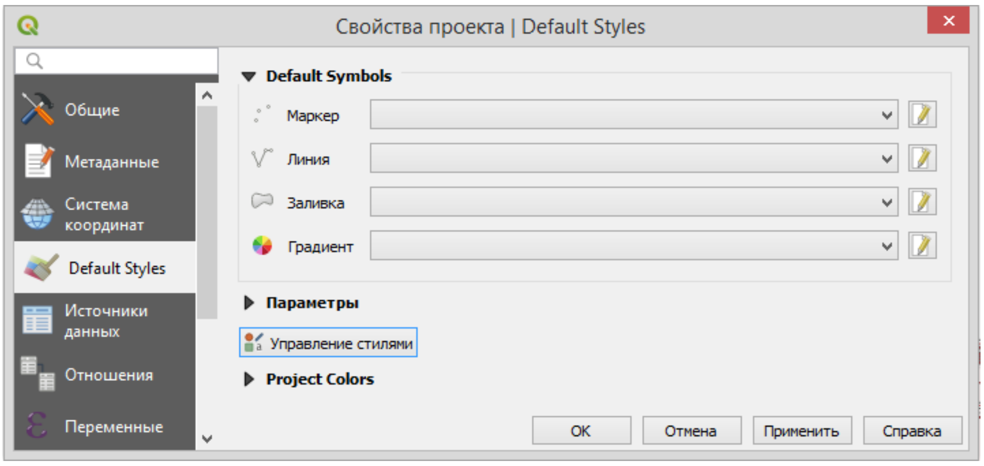
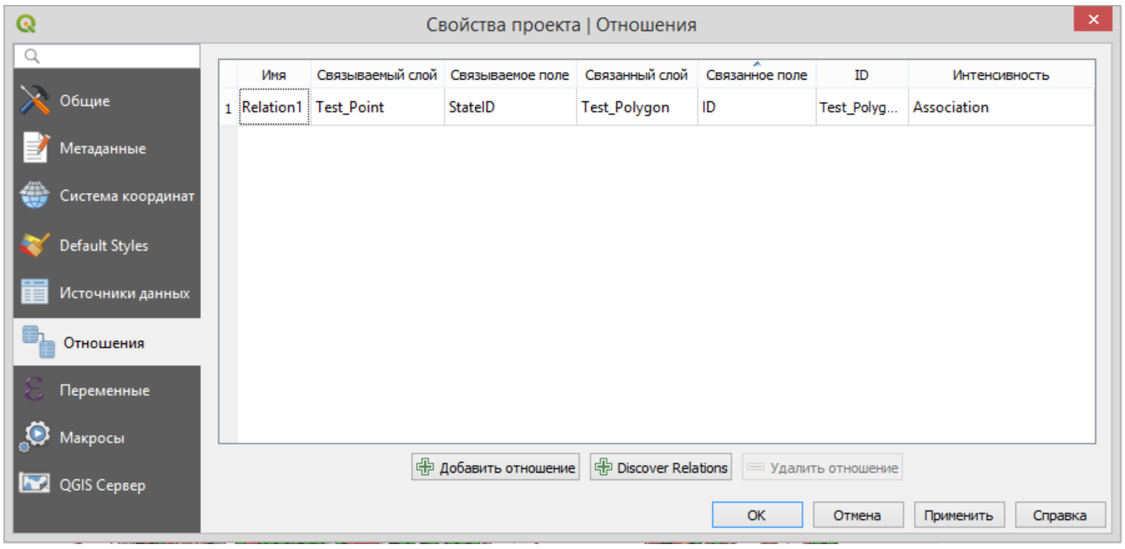


Figure 13 – Default styles for the project

## 3.4. Relations

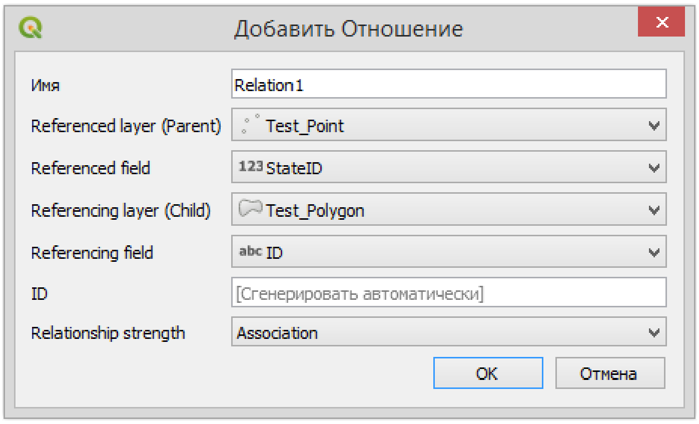
At the QGIS project level it is possible to set relations between layers and tables, that will be supported by publishing services in **eLiteGIS**, and then by building the map in **CoGIS**. At that no additional settings at the **eLiteGIS** or **CoGIS** levels will be required.

The relations between layers or tables should be set based on the values from the common attribute field, see Figure 14 and Figure 15.



| | Имя | Связываемый слой | Связываемое поле | Связанный слой | Связанное поле | ID | Интенсивность |
|---|---|---|---|---|---|---|---|
| 1 | Relation1 | Test_Point | StateID | Test_Polygon | ID | Test_Polyg... | Association |

Figure 14 – Relations between project's layers/tables



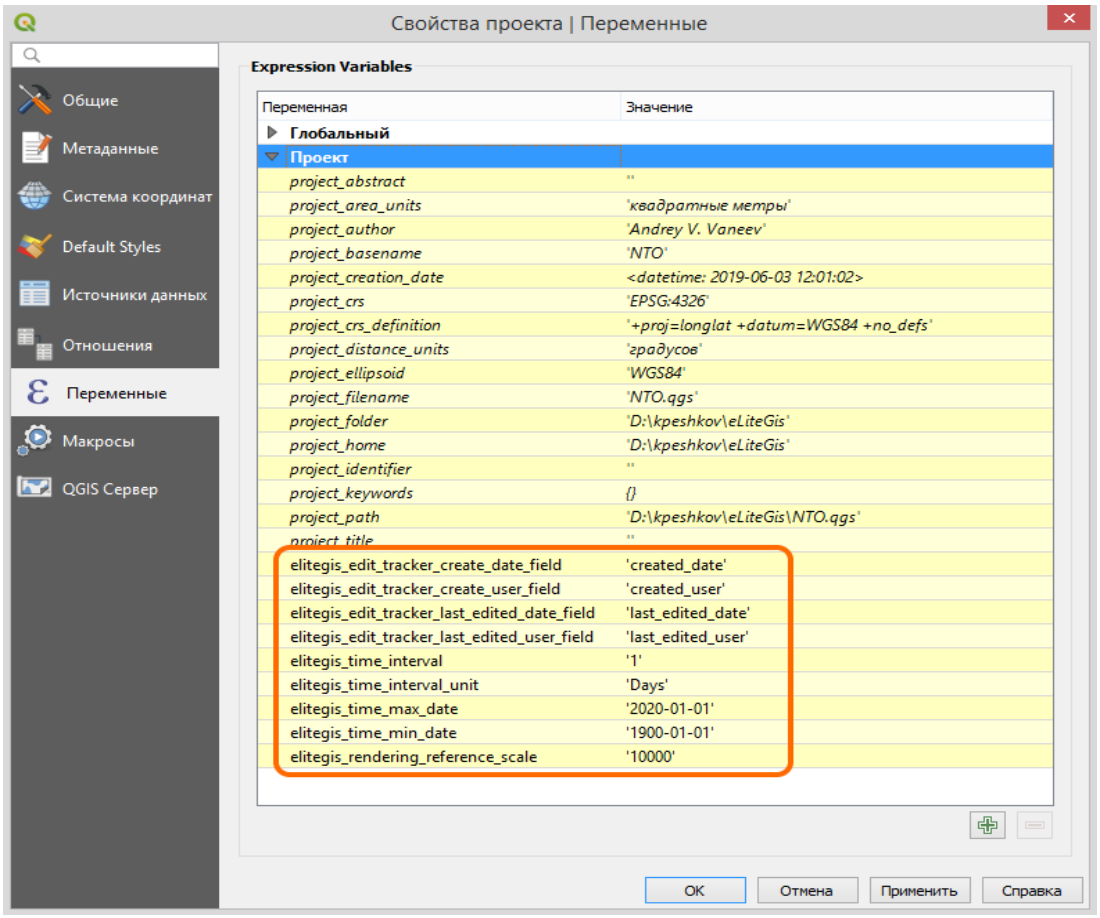
| | |
|---|---|
| Имя | Relation1 |
| Referenced layer (Parent) | Test_Point |
| Referenced field | 123 StateID |
| Referencing layer (Child) | Test_Polygon |
| Referencing field | abc ID |
| ID | [Сгенерировать автоматически] |
| Relationship strength | Association |

Figure 15 – Adding the new relation

## 3.5. Variables

At the QGIS project level it is possible to set variables that will be used by publishing services in **eLiteGIS**, for example, scale dependence of the symbology (*Scale Dependent*), reference scale (*Reference Scale*), edits tracking (*Edit Tracker*) and more (see section 6 for details).

The example of completed values for the project's variables is shown on Figure 16.



| Переменная | Значение |
|---|---|
| Глобальный | |
| Проект | |
| *project_abstract* | *''* |
| *project_area_units* | *'квадратные метры'* |
| *project_author* | *'Andrey V. Vaneev'* |
| *project_basename* | *'NTO'* |
| *project_creation_date* | *<datetime: 2019-06-03 12:01:02>* |
| *project_crs* | *'EPSG:4326'* |
| *project_crs_definition* | *'+proj=longlat +datum=WGS84 +no_defs'* |
| *project_distance_units* | *'градусов'* |
| *project_ellipsoid* | *'WGS84'* |
| *project_filename* | *'NTO.qgs'* |
| *project_folder* | *'D:\kpeshkov\eLiteGis'* |
| *project_home* | *'D:\kpeshkov\eLiteGis'* |
| *project_identifier* | *''* |
| *project_keywords* | *{}* |
| *project_path* | *'D:\kpeshkov\eLiteGis\NTO.qgs'* |
| *project_title* | *''* |
| elitegis_edit_tracker_create_date_field | 'created_date' |
| elitegis_edit_tracker_create_user_field | 'created_user' |
| elitegis_edit_tracker_last_edited_date_field | 'last_edited_date' |
| elitegis_edit_tracker_last_edited_user_field | 'last_edited_user' |
| elitegis_time_interval | '1' |
| elitegis_time_interval_unit | 'Days' |
| elitegis_time_max_date | '2020-01-01' |
| elitegis_time_min_date | '1900-01-01' |
| elitegis_rendering_reference_scale | '10000' |

Figure 16 – Setting variables at QGIS project level

# 4. Layers and tables

## 4.1. Feature layers

The feature layers are used for visualization of similar geographic/geometry vector features. The available geometry types of feature layers are point, polyline and polygon, see Figure 17.

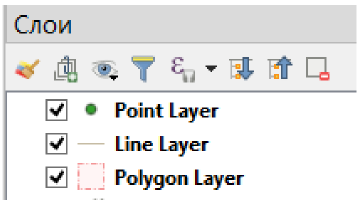


Figure 17 – Feature layers of different types

## 4.2. Raster layers

The raster layers are layers containing raster data only. For now, the raster layers created in QGIS project are not supported in **eLiteGIS**. But it is possible to publish the raster layers based on the GeoTIFF file directly via the **eLiteGIS** interface (web console **eLiteGIS Server Manager**, see **Publishing GIS services in eLiteGIS** for details).

## 4.3. Group layer

The group layer or complex layer is the layer containing sublayers of one or multiple types, see Figure 18.

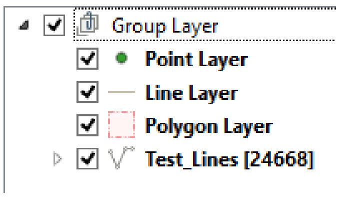
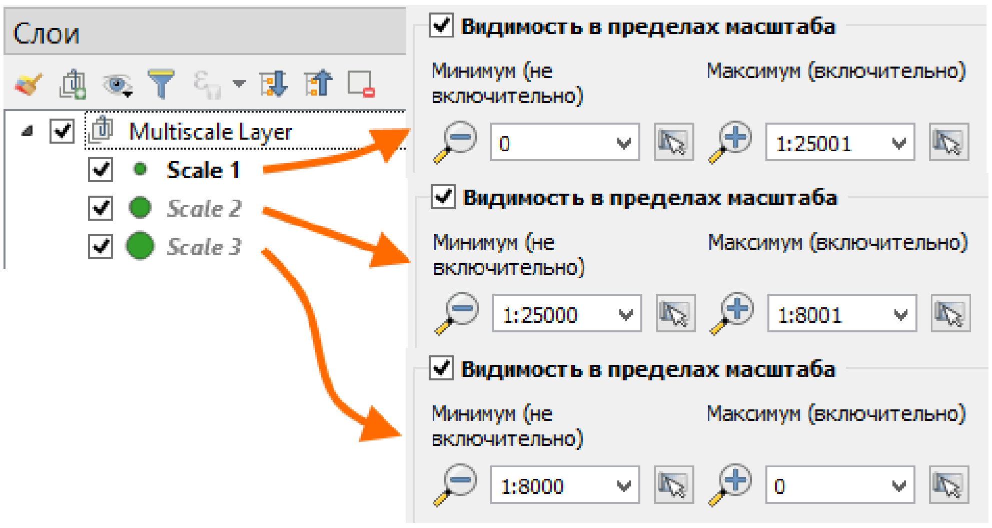


Figure 18 – Group layer

## 4.4. Multiscale layer

The multiscale layer is provided for visualization of the features representations in multiple scales. That is, with the multiscale layer the variable detailing of features depending on the map scale is set.

The example of setting the multiscale layer is shown on Figure 19.



Видимость в пределах масштаба
Минимум (не включительно) 0
Максимум (включительно) 1:25001

Видимость в пределах масштаба
Минимум (не включительно) 1:25000
Максимум (включительно) 1:8001

Видимость в пределах масштаба
Минимум (не включительно) 1:8000
Максимум (включительно) 0

Figure 19 – Multiscale layer

The multiscale layers can be of point, polyline or polygon type.

## 4.5.Tables

The table in the QGIS table of contents is the table data added to the project from the database. The table does not contain spatial data, but only the attribute data. The table needs to be added to the project (Table of contents window) for publishing the appropriate table data in **eLiteGIS**, see Figure 20

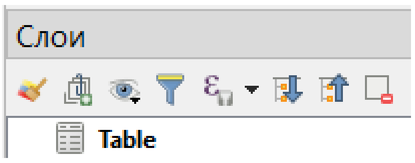


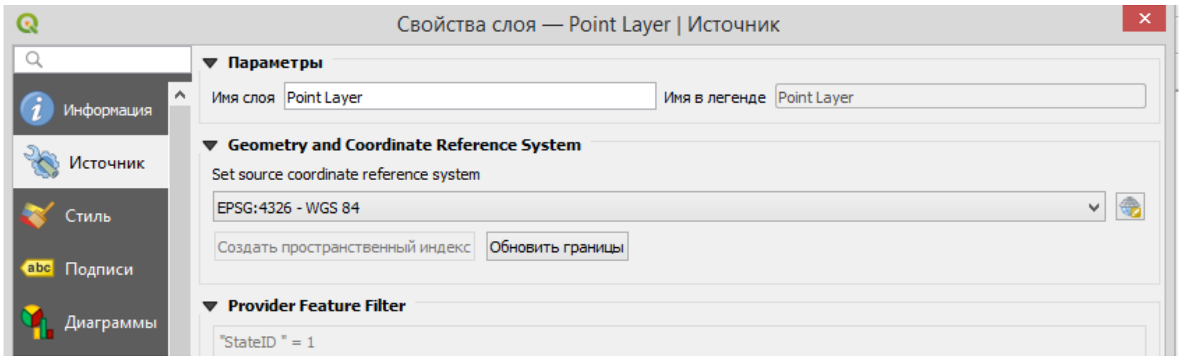
Figure 20 – The table in the list of layers of map project

# 5. Layer’s properties

## 5.1. Data source

### 5.1.1. Layer’s name

In the QGS project in the layer’s properties it is possible to specify or edit the layer’s name: Layer’s data source – Parameters - Layer’s name, see Figure 21.


Свойства слоя — Point Layer | Источник

Информация
Источник
Стиль
Подписи
Диаграммы

Параметры

Имя слоя: Point Layer
Имя в легенде: Point Layer

Geometry and Coordinate Reference System

Set source coordinate reference system: EPSG:4326 - WGS 84

Создать пространственный индекс
Обновить границы

Provider Feature Filter

"StateID " = 1

Figure 21 – Setting properties of the layer’s data source

### 5.1.2. Coordinate system

In the QGS project in the layer’s properties it is possible to specify or edit the coordinate system for specific layer: Layer’s data source - *Geometry and Coordinate Reference System - Set source coordinate reference system*, see Figure 21.

### 5.1.3. Definition query

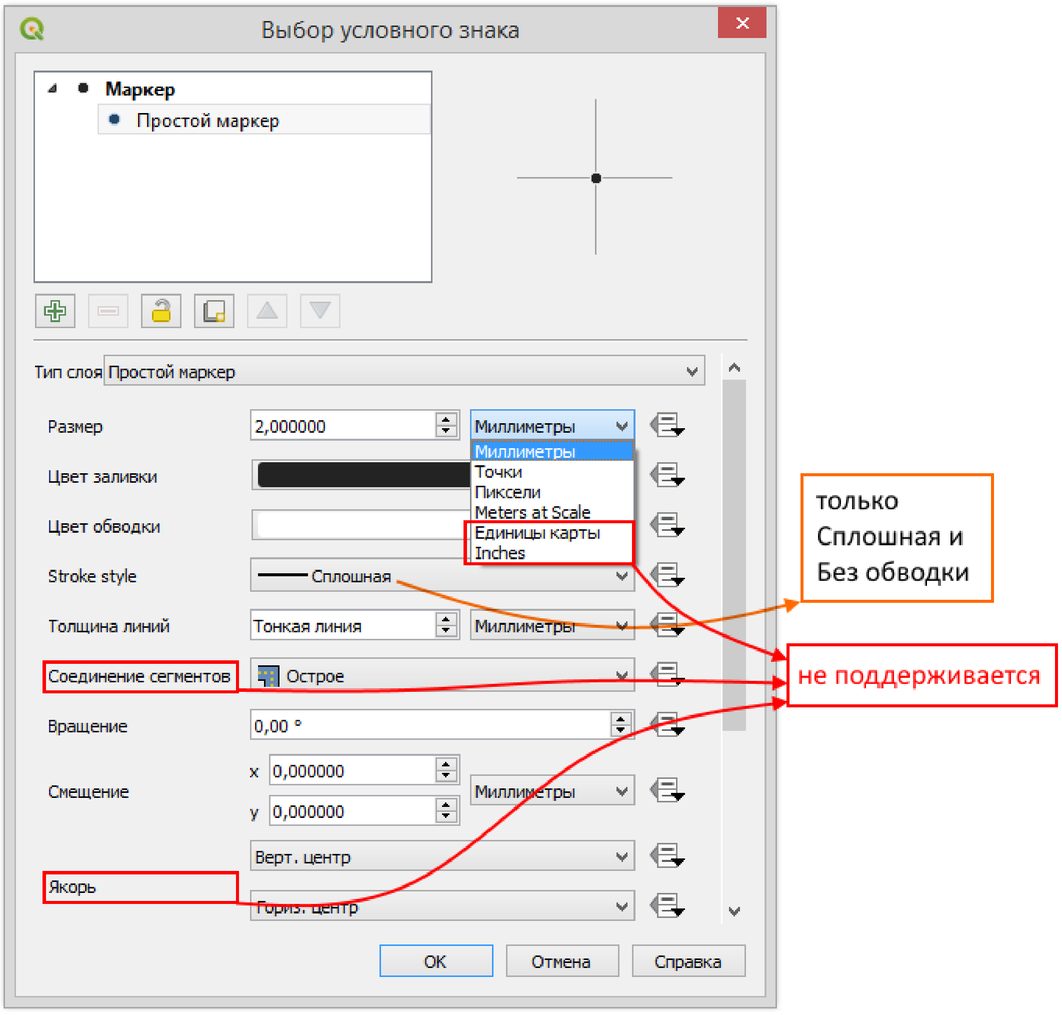
In the QGS project in the layer’s properties it is possible to set the SQL definition query for specific layer: Layer’s data source- *Provider Feature Filter*, see Figure 21.

## 5.2. Layer’s style

In the QGS project in the layer’s properties it is possible to specify the symbology to display layer’s features on the map that will be supported in the **eLiteGIS** services without additional settings.

### 5.2.1. Selecting layer’s symbol

For the layer it is possible to select the symbol, set its color, size and other parameters, see Figure 22. The figure shows which display parameters are supported by publishing the project as the map service in **eLiteGIS**.


Выбор условного знака

Маркер
Простой маркер

Тип слоя: Простой маркер

Размер: 2,000000 — Миллиметры (Миллиметры, Точки, Пиксели, Meters at Scale, Единицы карты, Inches)

Цвет заливки

Цвет обводки

Stroke style: Сплошная

Толщина линий: Тонкая линия — Миллиметры

Соединение сегментов: Острое

Вращение: 0,00 °

Смещение: x 0,000000, y 0,000000 — Миллиметры

Якорь: Верт. центр, Гориз. центр

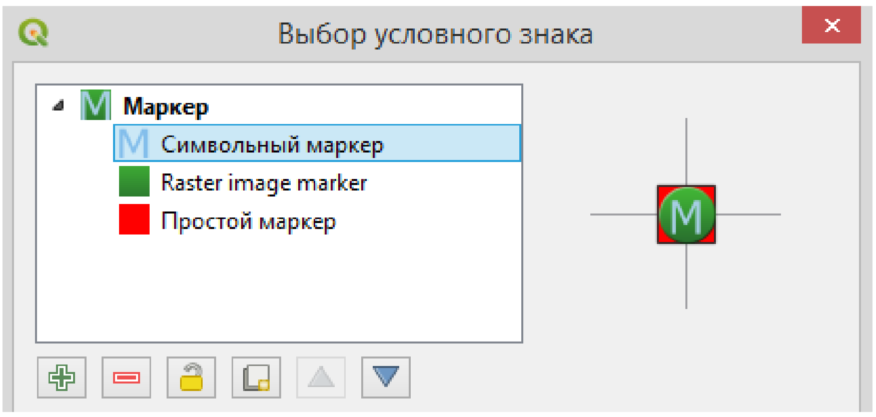
Figure 22 – Setting symbol for layer

### 5.2.2. Scale dependency

The size of symbols as scale dependent units is specified by selection of measurement units *Meters as scale*. For the size it is specified the quantity of meters in the reality for scale 1:1000 (*reference scale* in **eLiteGIS**).

### 5.2.3. Composite marker

For the layer, the possibility of building the symbol from multiple symbols, setting separate properties for each symbol is supported, see Figure 23.

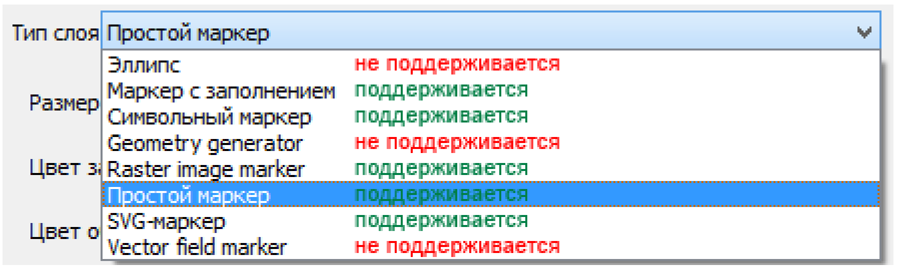

Выбор условного знака

Маркер
Символьный маркер
Raster image marker
Простой маркер

Figure 23 – Composite marker for layer

### 5.2.4. Types of symbols: marker

The figure below shows the types of symbols supported by publishing of map project as service in **eLiteGIS**, see Figure 24.



Тип слоя: Простой маркер

| | |
|---|---|
| Эллипс | не поддерживается |
| Маркер с заполнением | поддерживается |
| Символьный маркер | поддерживается |
| Geometry generator | не поддерживается |
| Raster image marker | поддерживается |
| Простой маркер | поддерживается |
| SVG-маркер | поддерживается |
| Vector field marker | не поддерживается |

Figure 24 – Types of symbols supported by publishing of map project as service in eLiteGIS

#### 5.2.4.1. Simple marker

The figure below shows the types of simple markers (simple shaped markers with customizable sizes and graphic properties) that are supported by publishing of map project as service in **eLiteGIS**, see Figure 25.

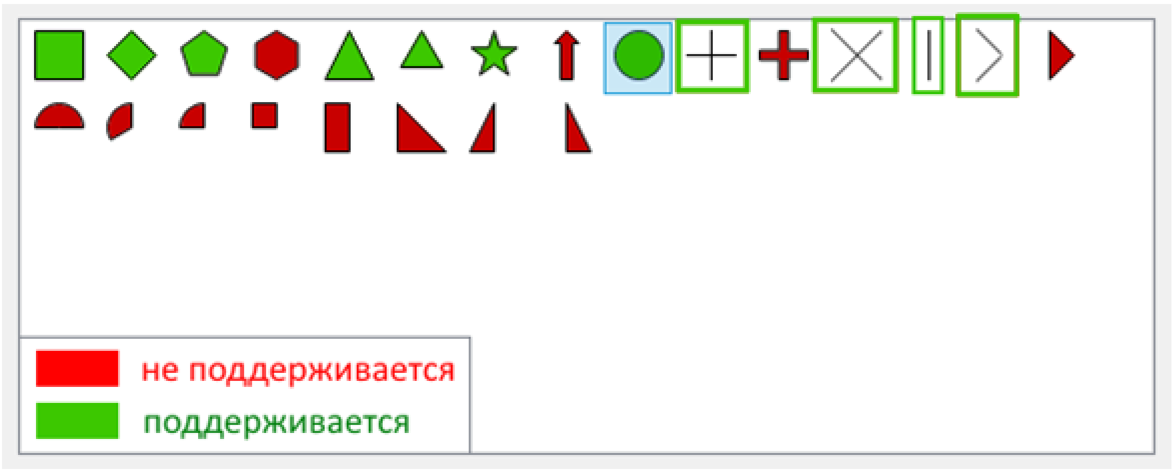


Figure 25 – Types of simple markers supported by publishing of map project as service in eLiteGIS

#### 5.2.4.2. Symbol marker

By publishing of map project as service in **eLiteGIS**, the use of symbol markers (markers created from symbols of the main fonts from the system folder) is supported, see Figure 26.

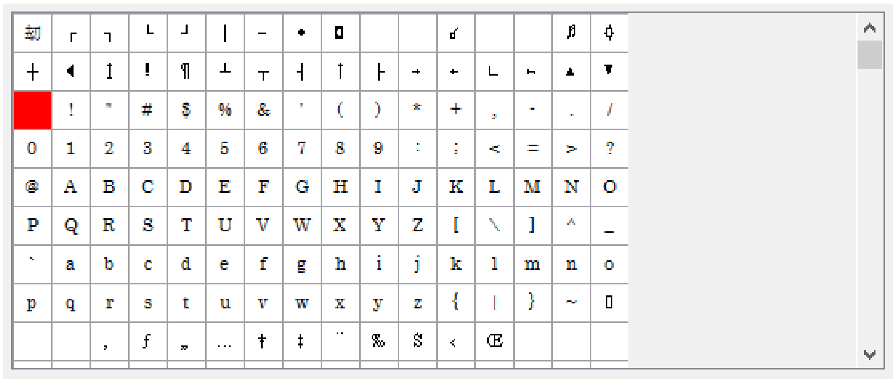


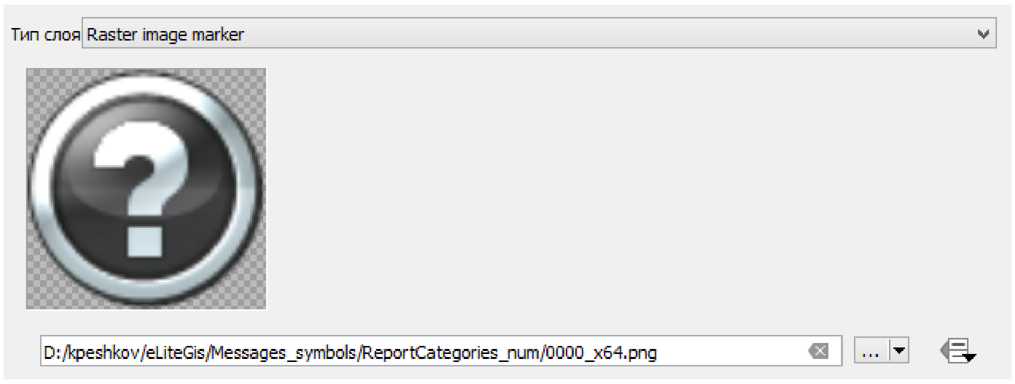
Figure 26 – Symbol marker

#### 5.2.4.3. Fill marker

By publishing of map project as service in **eLiteGIS**, the use of fill of simple marker based on the areal features rules is supported. All types of fill are supported, see section 5.2.6 for details.

#### 5.2.4.4. Marker – raster image

By publishing of map project as service in **eLiteGIS**, the use of marker symbol created from the graphic file of PNG (*.png), JPEG (*.jpg,*.jpeg), GIF (*.gif), Windows bitmap (.bmp) or Windows enhanced metafile (.emf) is supported, see Figure 27.



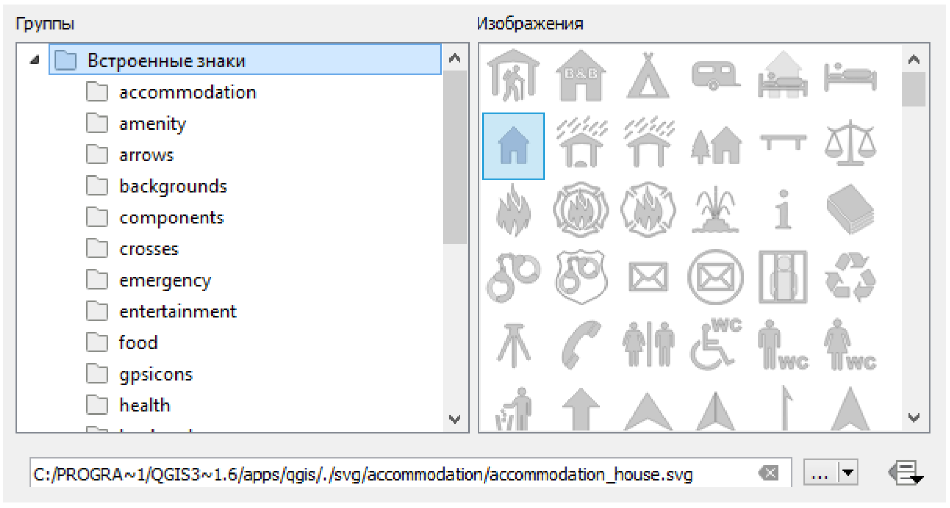
D:/kpeshkov/eLiteGis/Messages_symbols/ReportCategories_num/0000_x64.png

Figure 27 – Marker - raster image

*Note: Raster image marker saves the relative or absolute path to file depending on setting described in section 3.1 (it is recommended to save the relative path). The file should be copied to the server.*

#### 5.2.4.5. SVG marker

By publishing of map project as service in **eLiteGIS**, the use of SVG marker created based on graphic file in SVG (*.svg) format is supported, see Figure 28.

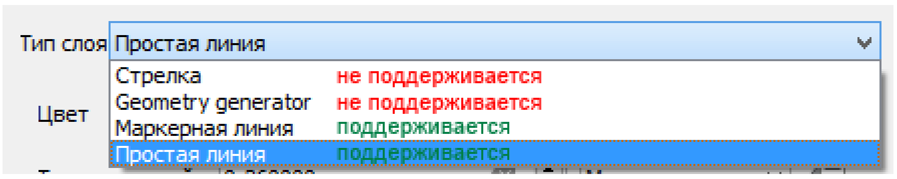


C:/PROGRA~1/QGIS3~1.6/apps/qgis/./svg/accommodation/accommodation_house.svg

Figure 28 – SVG marker

### 5.2.5. Types of symbols: polyline

The figure below shows the line symbology types that are supported when publishing the map project as a service in **eLiteGIS**, see Figure 29.

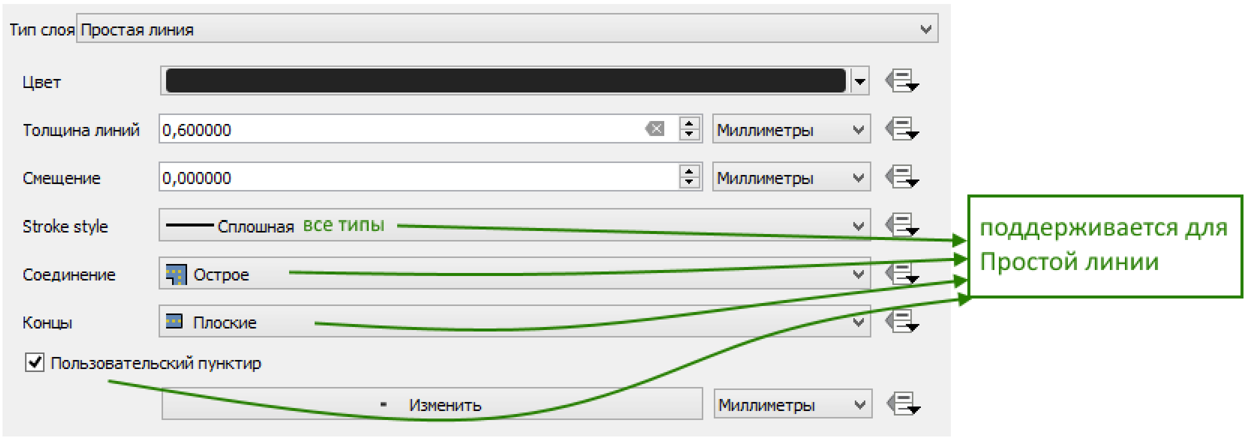

Тип слоя: Простая линия

| | |
|---|---|
| Стрелка | не поддерживается |
| Geometry generator | не поддерживается |
| Маркерная линия | поддерживается |
| Простая линия | поддерживается |

Figure 29 – Types of symbols for polylines supported by publishing of map project as service in eLiteGIS

### 5.2.5.1. Simple polyline

The figure below shows parameters for simple polyline supported by publishing of map project as service in **eLiteGIS**, see Figure 30.


Тип слоя: Простая линия

Цвет

Толщина линий: 0,600000 Миллиметры

Смещение: 0,000000 Миллиметры

Stroke style: Сплошная все типы

Соединение: Острое

Концы: Плоские

☑ Пользовательский пунктир

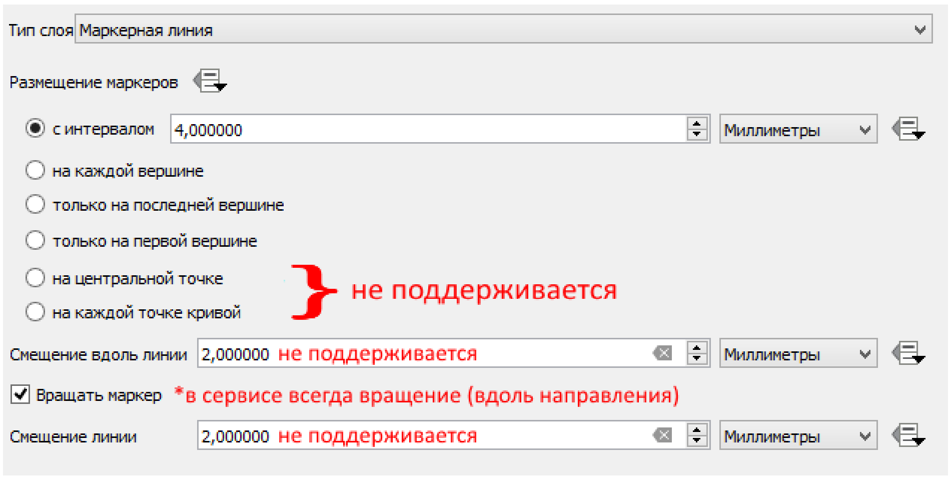
Изменить Миллиметры

поддерживается для Простой линии

Figure 30 – Simple polyline

### 5.2.5.2. Marker polyline

The figure below shows parameters for marker polyline supported by publishing of map project as service in **eLiteGIS**, see Figure 31.


Тип слоя: Маркерная линия

Размещение маркеров

● с интервалом 4,000000 Миллиметры

○ на каждой вершине

○ только на последней вершине

○ только на первой вершине

○ на центральной точке

○ на каждой точке кривой

не поддерживается

Смещение вдоль линии: 2,000000 не поддерживается Миллиметры

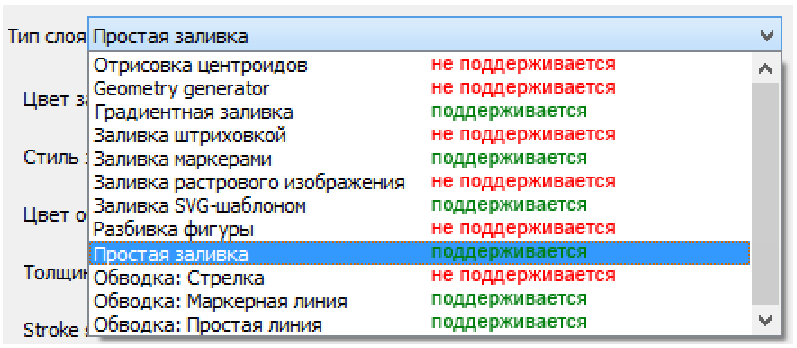
☑ Вращать маркер *в сервисе всегда вращение (вдоль направления)

Смещение линии: 2,000000 не поддерживается Миллиметры

Figure 31 – Marker polyline

## 5.2.6. Types of symbols: fill

The figure below shows types of fill supported by publishing of map project as service in **eLiteGIS**, see Figure 32.

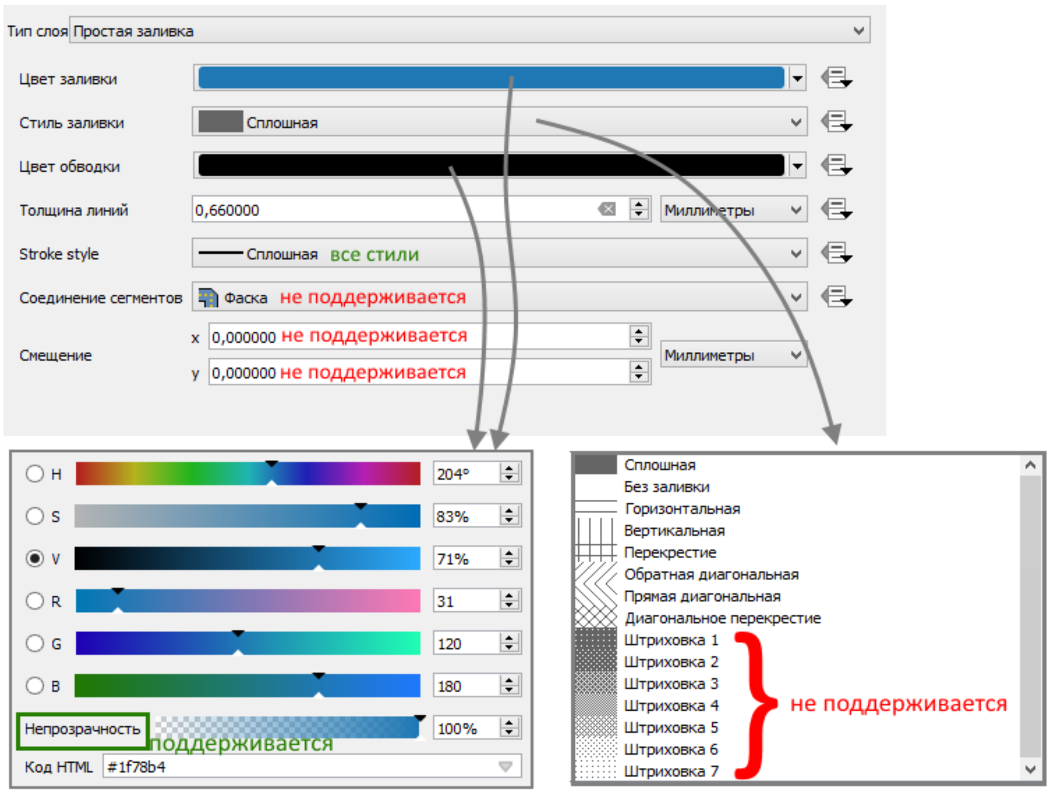

Тип слоя Простая заливка

| | |
|---|---|
| Отрисовка центроидов | не поддерживается |
| Geometry generator | не поддерживается |
| Градиентная заливка | поддерживается |
| Заливка штриховкой | не поддерживается |
| Заливка маркерами | поддерживается |
| Заливка растрового изображения | не поддерживается |
| Заливка SVG-шаблоном | поддерживается |
| Разбивка фигуры | не поддерживается |
| Простая заливка | поддерживается |
| Обводка: Стрелка | не поддерживается |
| Обводка: Маркерная линия | поддерживается |
| Обводка: Простая линия | поддерживается |

Figure 32 – Types of fill supported by publishing of map project as service in eLiteGIS

5.2.6.1. Simple fill

The figure below shows parameters for simple fill supported by publishing of map project as service in **eLiteGIS**, see Figure 33.


Тип слоя Простая заливка

| | |
|---|---|
| Цвет заливки | |
| Стиль заливки | Сплошная |
| Цвет обводки | |
| Толщина линий | 0,660000 Миллиметры |
| Stroke style | Сплошная все стили |
| Соединение сегментов | Фаска не поддерживается |
| Смещение | x 0,000000 не поддерживается; y 0,000000 не поддерживается; Миллиметры |

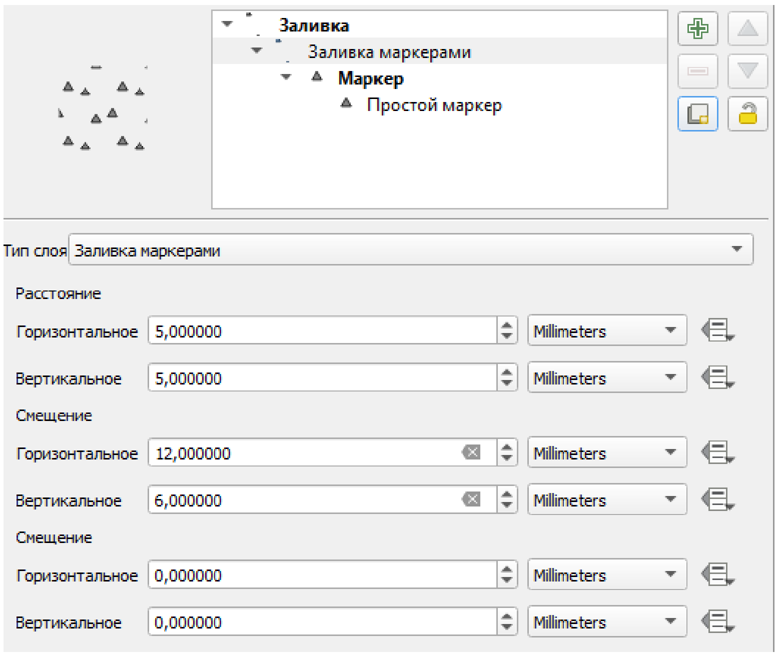
| | |
|---|---|
| H | 204° |
| S | 83% |
| V | 71% |
| R | 31 |
| G | 120 |
| B | 180 |
| Непрозрачность | 100% поддерживается |
| Код HTML | #1f78b4 |

Сплошная
Без заливки
Горизонтальная
Вертикальная
Перекрестие
Обратная диагональная
Прямая диагональная
Диагональное перекрестие
Штриховка 1
Штриховка 2
Штриховка 3
Штриховка 4
Штриховка 5
Штриховка 6
Штриховка 7
$\}$ не поддерживается

Figure 33 – Simple fill

5.2.6.2. Marker fill

By publishing of map project as service in **eLiteGIS**, the marker fill is supported, see Figure 34.



Заливка
Заливка маркерами
Маркер
Простой маркер

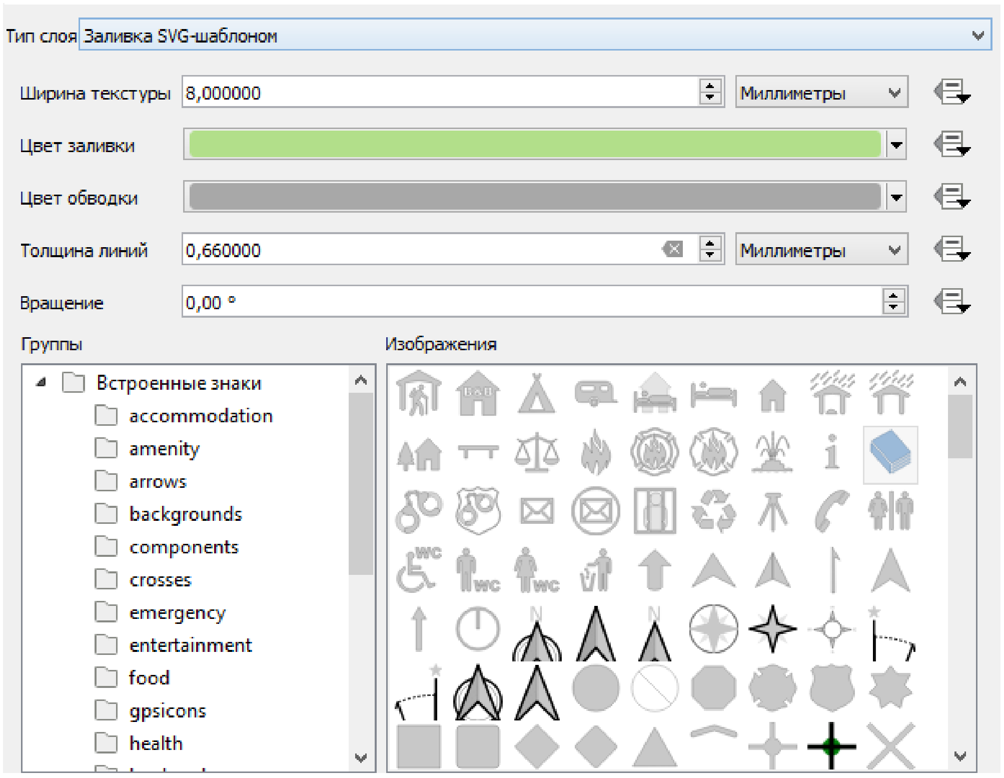
Тип слоя Заливка маркерами

Расстояние
Горизонтальное 5,000000 Millimeters
Вертикальное 5,000000 Millimeters
Смещение
Горизонтальное 12,000000 Millimeters
Вертикальное 6,000000 Millimeters
Смещение
Горизонтальное 0,000000 Millimeters
Вертикальное 0,000000 Millimeters

Figure 34 – Marker fill

5.2.6.3. SVG template fill

By publishing of map project as service in **eLiteGIS**, the SVG template fill is supported, see Figure 35.



Тип слоя Заливка SVG-шаблоном

Ширина текстуры 8,000000 Миллиметры
Цвет заливки
Цвет обводки
Толщина линий 0,660000 Миллиметры
Вращение 0,00 °

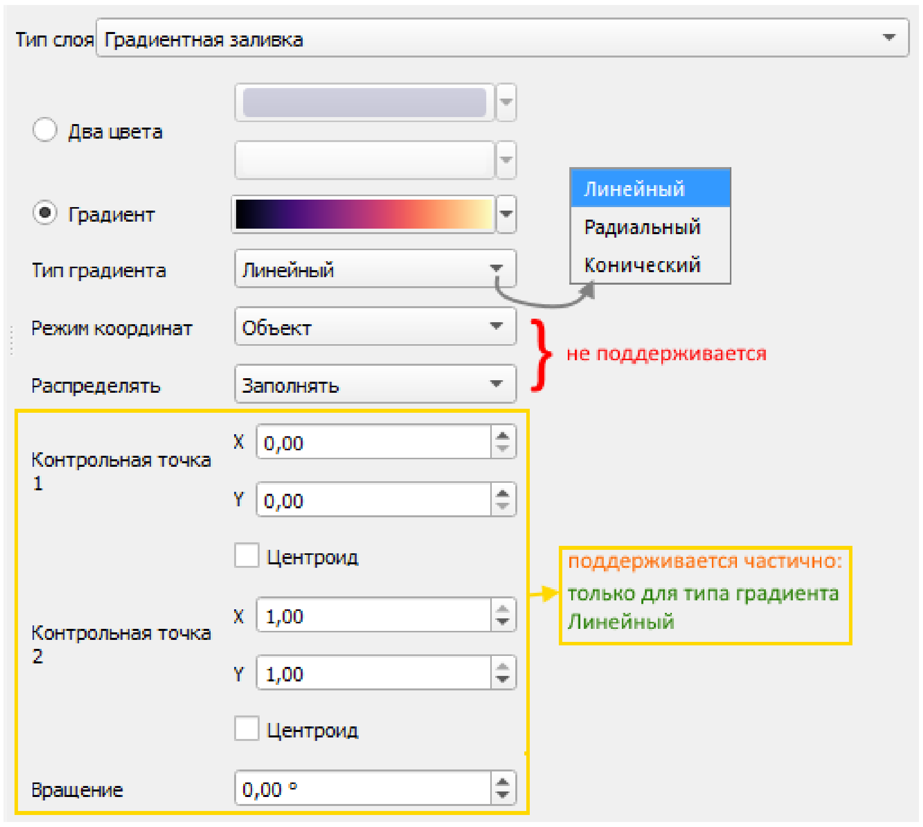
Группы
Встроенные знаки
accommodation
amenity
arrows
backgrounds
components
crosses
emergency
entertainment
food
gpsicons
health

Изображения

Figure 35 – SVG template fill

5.2.6.4. Gradient fill

By publishing of map project as service in **eLiteGIS**, the gradient fill is supported, see Figure 36.


Тип слоя Градиентная заливка

Два цвета

Градиент

Тип градиента Линейный

Режим координат Объект

Распределять Заполнять

Контрольная точка 1 X 0,00 Y 0,00 Центроид

Контрольная точка 2 X 1,00 Y 1,00 Центроид

Вращение 0,00 °

Figure 36 – Gradient fill

5.2.6.5. Outline: marker polyline

The settings of this outline are similar to settings of the marker polyline, see section 5.2.5.

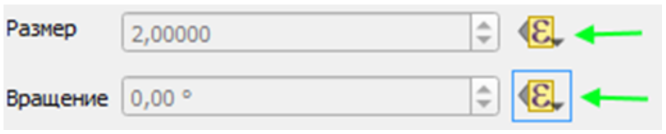
5.2.6.6. Outline: simple polyline

The settings of this outline are similar to settings of simple polyline, see section 5.2.5.

### 5.2.7. Redefining data (Data defined override)

**eLiteGIS** supports symbology and labeling parameters specified at the QGIS project level based on values of selected fields or SQL expressions.

The ability of data redefinition is supported for such parameters as size, rotation angle, see Figure 37.


Размер 2,00000

Вращение 0,00 °

Figure 37 – Setting size and rotation angle

### 5.2.8. List of functions supported in SQL expressions (Expression Dialog в QGIS)

**eLiteGIS** supports SQL expressions specified at the QGIS project level using the following functions: *"sin", "cos", "tan", "atan", "abs", "asin", "acos", "log", "log10", "cailing", "floor", "round", "ltrim", "rtrim", "substr", "substring", "concat", "lower", "upper", "pow", "andbits", "len", "length","coalesce", "mod", "scale_linear", "scale_exp", "tostring"*, see Figure 38.

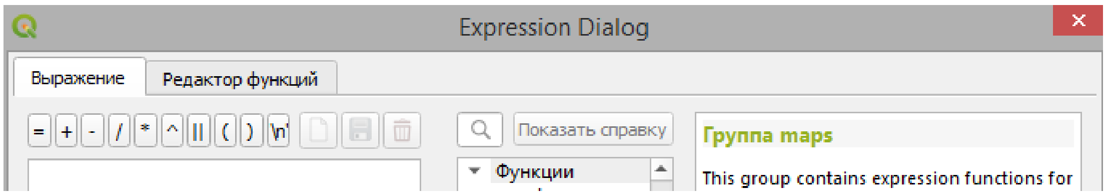


Figure 38 – SQL expressions specified at the QGIS project level

### 5.2.9. General symbology settings

The below figure shows general settings of symbology supported by publishing of map project as service in **eLiteGIS**, see Figure 39.

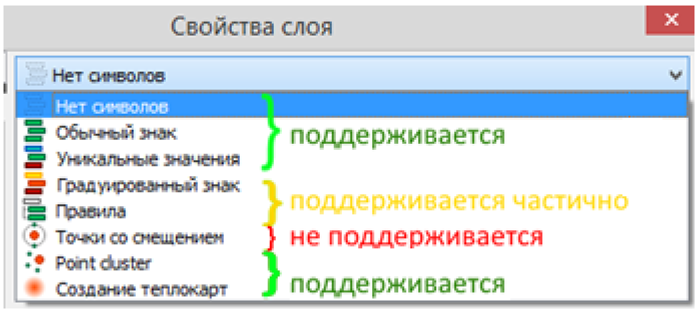


Figure 39 – General settings of symbology supported by publishing of map project as service in eLiteGIS

#### 5.2.9.1. Symbology setting: no symbols

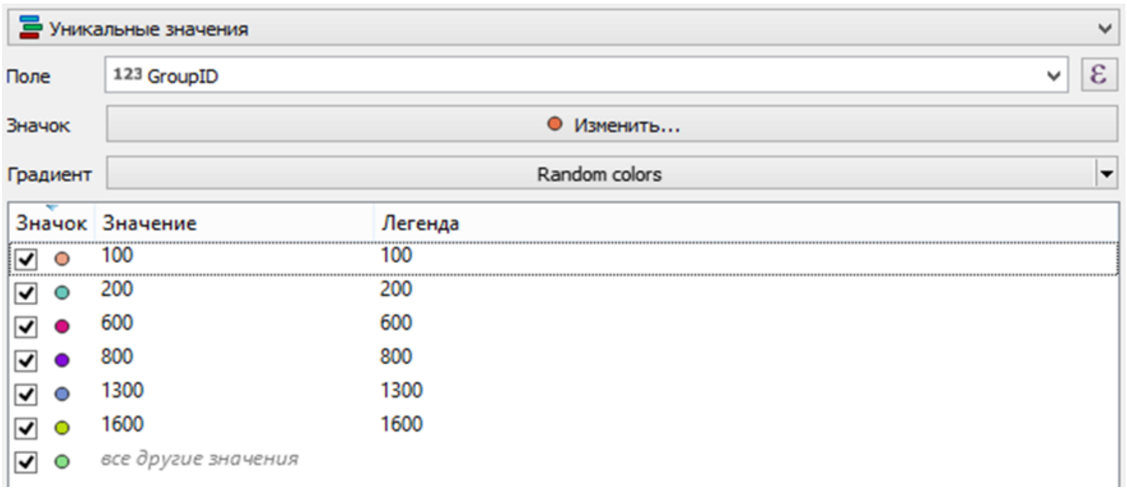
The symbols for features of this layer will not be displayed.

#### 5.2.9.2. Symbology setting: common symbol

All features of the layers will be displayed with the common symbol.

#### 5.2.9.3. Symbology setting: unique values

The symbols will be displayed based on the unique value of the selected field of the feature, see Figure 40.



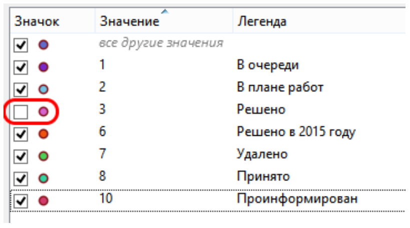
Уникальные значения

| | |
|---|---|
| Поле | 123 GroupID |
| Значок | Изменить... |
| Градиент | Random colors |

| Значок | Значение | Легенда |
|---|---|---|
| ✓ ● | 100 | 100 |
| ✓ ● | 200 | 200 |
| ✓ ● | 600 | 600 |
| ✓ ● | 800 | 800 |
| ✓ ● | 1300 | 1300 |
| ✓ ● | 1600 | 1600 |
| ✓ ● | *все другие значения* | |

Figure 40 – Symbology setting: unique values

At that, if you turn off visibility of one of the values in QGIS project, see Figure 41, then in the published service in **eLiteGIS** and **CoGIS** this value will not be displayed in the legend and on the map. But the label will be displayed if it has been specified. Besides, you will be able to identify feature on the map. And, if style for category *all other values* has been set, the features from the category of turned off style will get to this category.



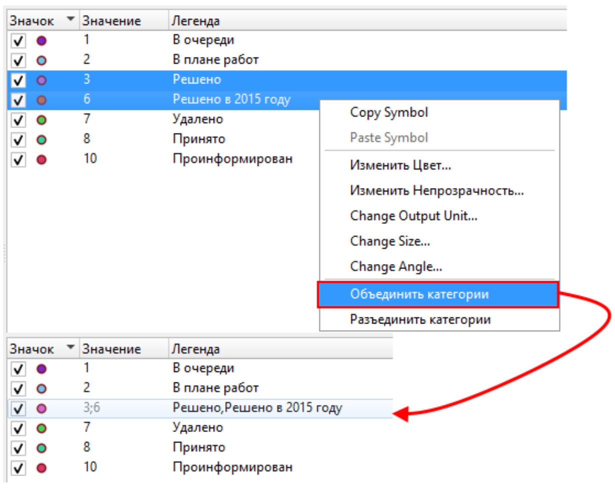
| Значок | Значение | Легенда |
|---|---|---|
| ✓ ● | *все другие значения* | |
| ✓ ● | 1 | В очереди |
| ✓ ● | 2 | В плане работ |
| ☐ ● | 3 | Решено |
| ✓ ● | 6 | Решено в 2015 году |
| ✓ ● | 7 | Удалено |
| ✓ ● | 8 | Принято |
| ✓ ● | 10 | Проинформирован |

Figure 41 – Turning off visibility of one of the values at the project level

*Note: We do not recommend turning off specific style in the project. If you do not need the style, just do not specify it.*

When the style for *all other values* is specified, all features that have not been described separately would get to this category. For this style it will be impossible to turn off visibility in the legend.

With the Combined categories setting it is possible to combine multiple categories to one, see Figure 42.



| Значок | Значение | Легенда |
|---|---|---|
| ✓ ● | 1 | В очереди |
| ✓ ● | 2 | В плане работ |
| ✓ ● | 3 | Решено |
| ✓ ● | 6 | Решено в 2015 году |
| ✓ ● | 7 | Удалено |
| ✓ ● | 8 | Принято |
| ✓ ● | 10 | Проинформирован |

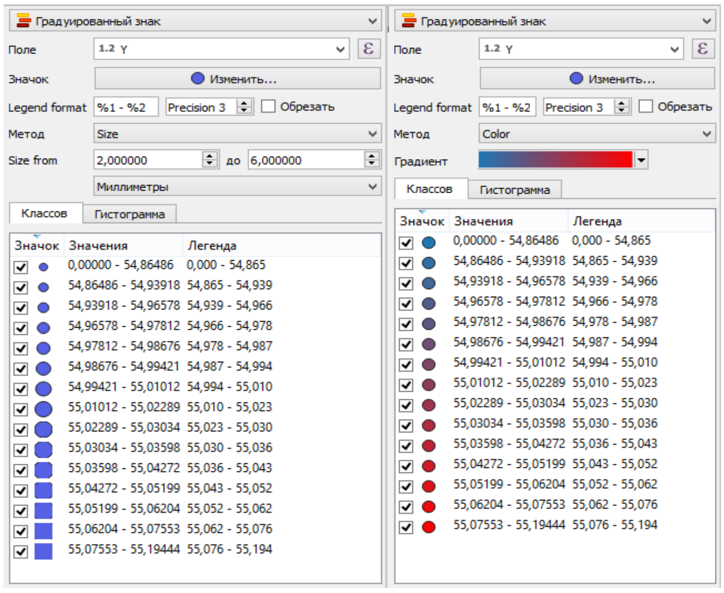
| Значок | Значение | Легенда |
|---|---|---|
| ✓ ● | 1 | В очереди |
| ✓ ● | 2 | В плане работ |
| ✓ ● | 3;6 | Решено,Решено в 2015 году |
| ✓ ● | 7 | Удалено |
| ✓ ● | 8 | Принято |
| ✓ ● | 10 | Проинформирован |

Figure 42 – Combined categories for unique values

### 5.2.9.4. Symbol setting: graduated symbol

The graduated symbol is set for displaying the quantitative differences between mapped featured via changing the symbols color or size. The data is classified by ranges, with specific color or size for each range, Figure 43.

The graduated symbol is supported with the limitation: for such type of visualization at the level of the web map in CoGIS it will not be possible to set calculation of features by each symbology.



Градуированный знак

Поле: 1.2 Y

Значок: Изменить...

Legend format: %1 - %2 | Precision 3 | ☐ Обрезать

Метод: Size

Size from: 2,000000 до 6,000000

Миллиметры

Классов | Гистограмма

| Значок | Значения | Легенда |
|---|---|---|
| ✓ ● | 0,00000 - 54,86486 | 0,000 - 54,865 |
| ✓ ● | 54,86486 - 54,93918 | 54,865 - 54,939 |
| ✓ ● | 54,93918 - 54,96578 | 54,939 - 54,966 |
| ✓ ● | 54,96578 - 54,97812 | 54,966 - 54,978 |
| ✓ ● | 54,97812 - 54,98676 | 54,978 - 54,987 |
| ✓ ● | 54,98676 - 54,99421 | 54,987 - 54,994 |
| ✓ ● | 54,99421 - 55,01012 | 54,994 - 55,010 |
| ✓ ● | 55,01012 - 55,02289 | 55,010 - 55,023 |
| ✓ ● | 55,02289 - 55,03034 | 55,023 - 55,030 |
| ✓ ● | 55,03034 - 55,03598 | 55,030 - 55,036 |
| ✓ ■ | 55,03598 - 55,04272 | 55,036 - 55,043 |
| ✓ ■ | 55,04272 - 55,05199 | 55,043 - 55,052 |
| ✓ ■ | 55,05199 - 55,06204 | 55,052 - 55,062 |
| ✓ ■ | 55,06204 - 55,07553 | 55,062 - 55,076 |
| ✓ ■ | 55,07553 - 55,19444 | 55,076 - 55,194 |

Градуированный знак

Поле: 1.2 Y

Значок: Изменить...

Legend format: %1 - %2 | Precision 3 | ☐ Обрезать

Метод: Color

Градиент

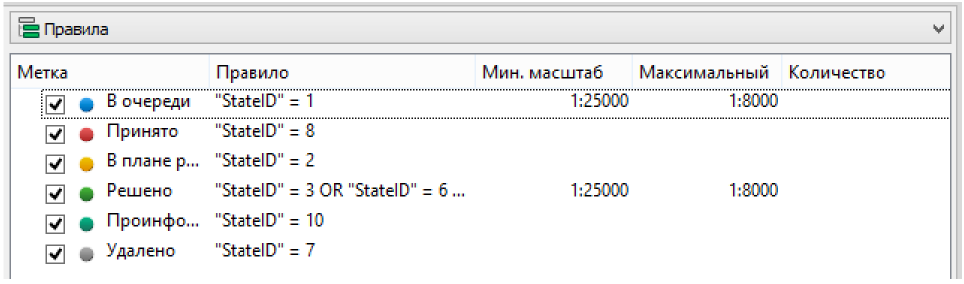
Классов | Гистограмма

| Значок | Значения | Легенда |
|---|---|---|
| ✓ ● | 0,00000 - 54,86486 | 0,000 - 54,865 |
| ✓ ● | 54,86486 - 54,93918 | 54,865 - 54,939 |
| ✓ ● | 54,93918 - 54,96578 | 54,939 - 54,966 |
| ✓ ● | 54,96578 - 54,97812 | 54,966 - 54,978 |
| ✓ ● | 54,97812 - 54,98676 | 54,978 - 54,987 |
| ✓ ● | 54,98676 - 54,99421 | 54,987 - 54,994 |
| ✓ ● | 54,99421 - 55,01012 | 54,994 - 55,010 |
| ✓ ● | 55,01012 - 55,02289 | 55,010 - 55,023 |
| ✓ ● | 55,02289 - 55,03034 | 55,023 - 55,030 |
| ✓ ● | 55,03034 - 55,03598 | 55,030 - 55,036 |
| ✓ ● | 55,03598 - 55,04272 | 55,036 - 55,043 |
| ✓ ● | 55,04272 - 55,05199 | 55,043 - 55,052 |
| ✓ ● | 55,05199 - 55,06204 | 55,052 - 55,062 |
| ✓ ● | 55,06204 - 55,07553 | 55,062 - 55,076 |
| ✓ ● | 55,07553 - 55,19444 | 55,076 - 55,194 |

Figure 43 – Graduated symbol

### 5.2.9.5. Symbol setting: rules

**eLiteGIS** supports symbols set for the layer in the QGIS project via the SQL filter, see Figure 44.

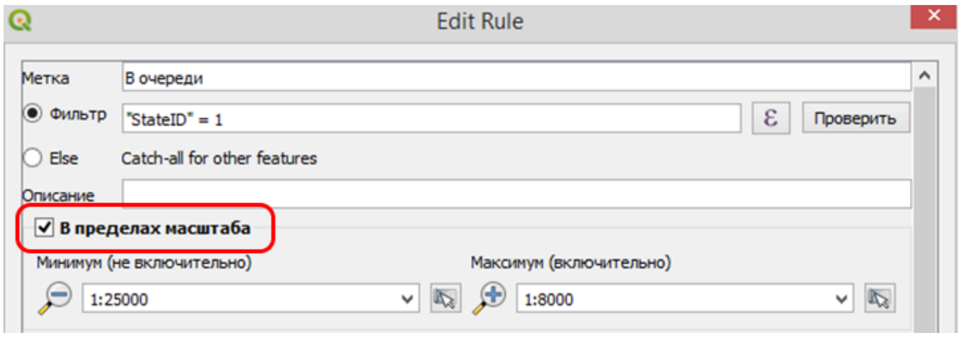


Правила

| Метка | Правило | Мин. масштаб | Максимальный | Количество |
|---|---|---|---|---|
| ✓ ● В очереди | "StateID" = 1 | 1:25000 | 1:8000 | |
| ✓ ● Принято | "StateID" = 8 | | | |
| ✓ ● В плане р... | "StateID" = 2 | | | |
| ✓ ● Решено | "StateID" = 3 OR "StateID" = 6 ... | 1:25000 | 1:8000 | |
| ✓ ● Проинфо... | "StateID" = 10 | | | |
| ✓ ● Удалено | "StateID" = 7 | | | |

Figure 44 – SQL filters for layer display

SQL filter is supported with the limitation: for such type of visualization at the level of the web map in **CoGIS** it will not be possible to set calculation of features by each symbology. The visibility within the scale is also not supported in the rules editor, see Figure 45.



Edit Rule

Метка: В очереди

● Фильтр: "StateID" = 1 | Проверить

○ Else: Catch-all for other features

Описание:

☑ **В пределах масштаба**

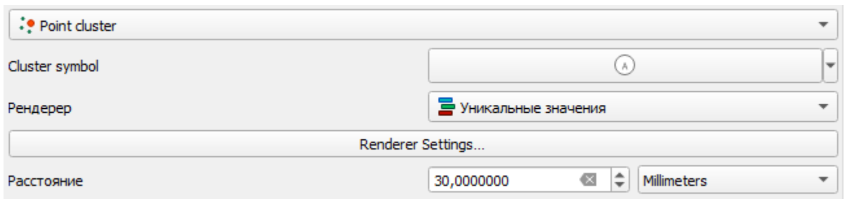
Минимум (не включительно): 1:25000

Максимум (включительно): 1:8000

Figure 45 – Visibility within the scale

### 5.2.9.6. Symbol setting: clusterization

**eLiteGIS** supports the options of features grouping and display of group in the group centroid specified at the level of QGIS project, see Figure 46.



Point cluster

| | |
|---|---|
| Cluster symbol | |
| Рендерер | Уникальные значения |
| Renderer Settings... | |
| Расстояние | 30,0000000 Millimeters |

Figure 46 – Clusterization

Learn more about setting clusterization and additional properties in section 6 below.

#### 5.2.9.7. Symbol setting: creation of heat maps

The heat maps are used for visualizing clusters of point data and identifying high concentrations of activity, see Figure 47.

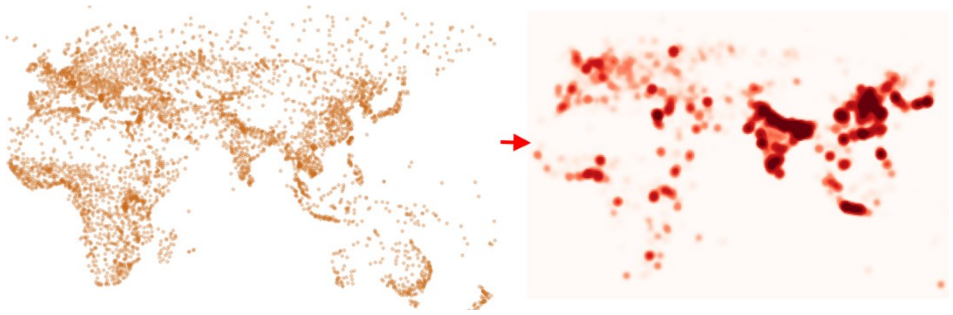
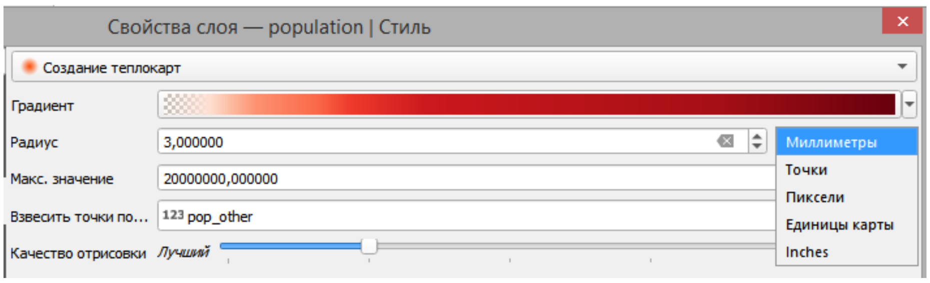


Figure 47 – Example of heat map

**eLiteGIS** supports layer symbology as heat maps and interpolation maps set at the level of QGIS project, see Figure 48.



Свойства слоя — population | Стиль

Создание теплокарт

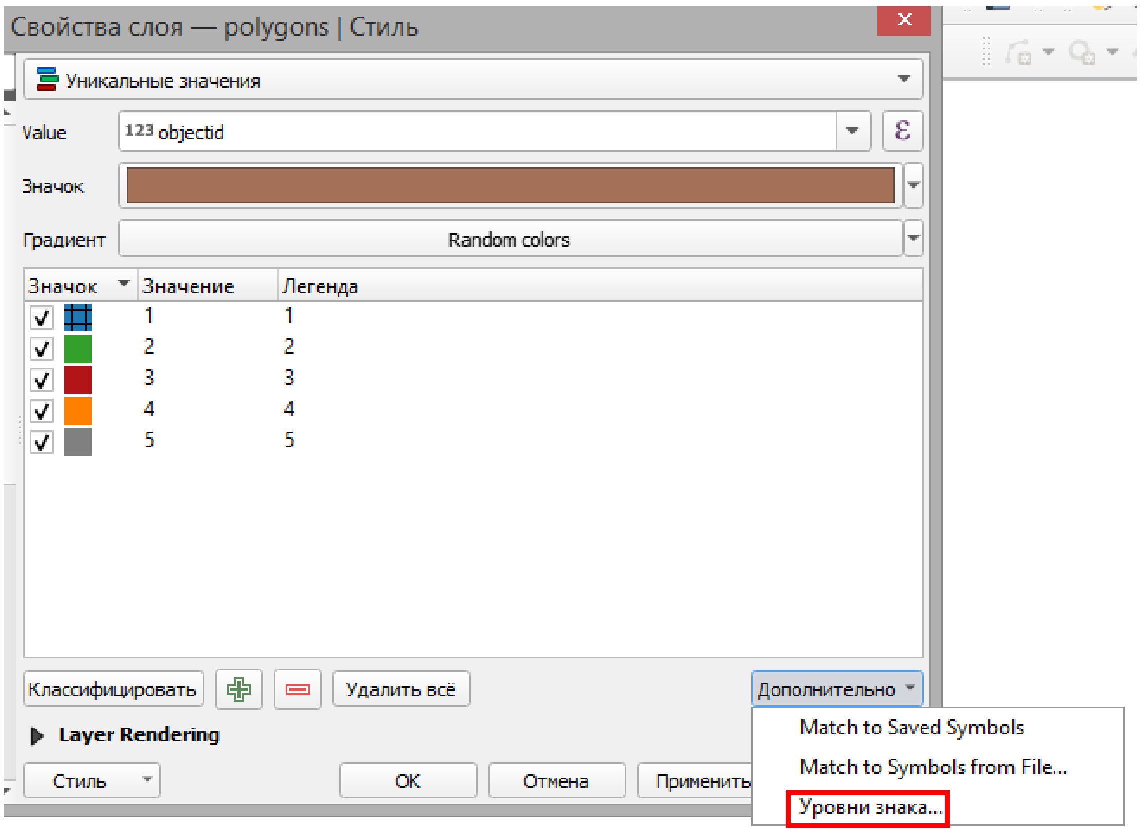
| | |
|---|---|
| Градиент | |
| Радиус | 3,000000 Миллиметры |
| Макс. значение | 20000000,000000 |
| Взвесить точки по... | 123 pop_other |
| Качество отрисовки | Лучший |

Figure 48 – Setting layer symbology as heat map

For additional setting of interpolation maps display in **eLiteGIS**, you can set properties in the variables settings, see section 6.9 for more details.

#### 5.2.9.8. Symbol levels

The symbol levels are used for setting the order of the symbology rendering: Unique values, Graduated symbol, Rules, see Figure 49.


Свойства слоя — polygons | Стиль

Уникальные значения

Value: 123 objectid

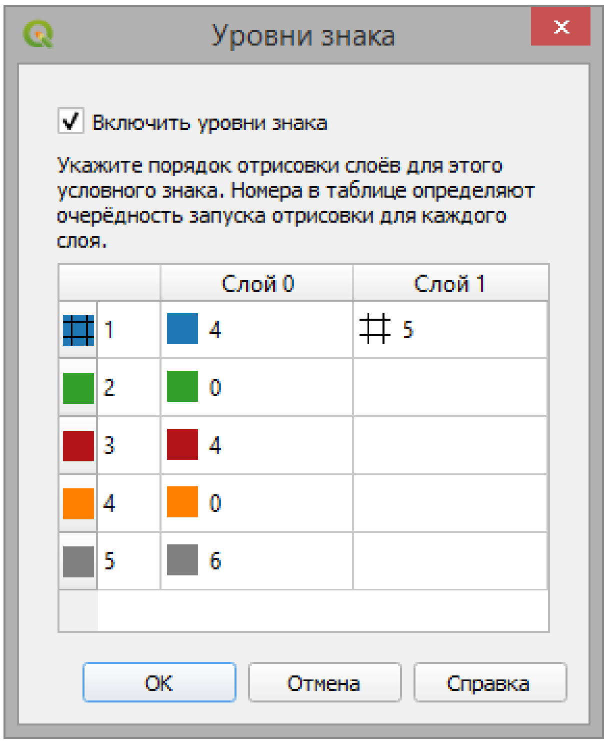
Значок

Градиент: Random colors

| Значок | Значение | Легенда |
|---|---|---|
| ✓ ▦ | 1 | 1 |
| ✓ 🟩 | 2 | 2 |
| ✓ 🟥 | 3 | 3 |
| ✓ 🟧 | 4 | 4 |
| ✓ ◼ | 5 | 5 |

Figure 49 – Symbol levels (1)

The order of rendering can be set not only for unique values, but for all layers of the symbology, thus, different symbol layers are mixed in various unique values, see Figure 50.


Уровни знака

✓ Включить уровни знака

Укажите порядок отрисовки слоёв для этого условного знака. Номера в таблице определяют очерёдность запуска отрисовки для каждого слоя.

| | Слой 0 | Слой 1 |
|---|---|---|
| ▦ 1 | 🟦 4 | # 5 |
| 🟩 2 | 🟩 0 | |
| 🟥 3 | 🟥 4 | |
| 🟧 4 | 🟧 0 | |
| ◼ 5 | ◼ 6 | |

Figure 50 – Symbol levels (2)

### 5.2.10. Layer rendering

**eLiteGIS** supports some parameters of layer rendering set at the level of map project in QGIS, see Figure 51.

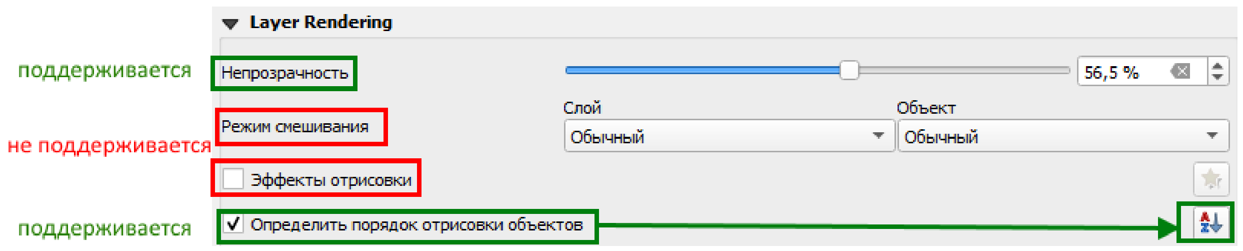
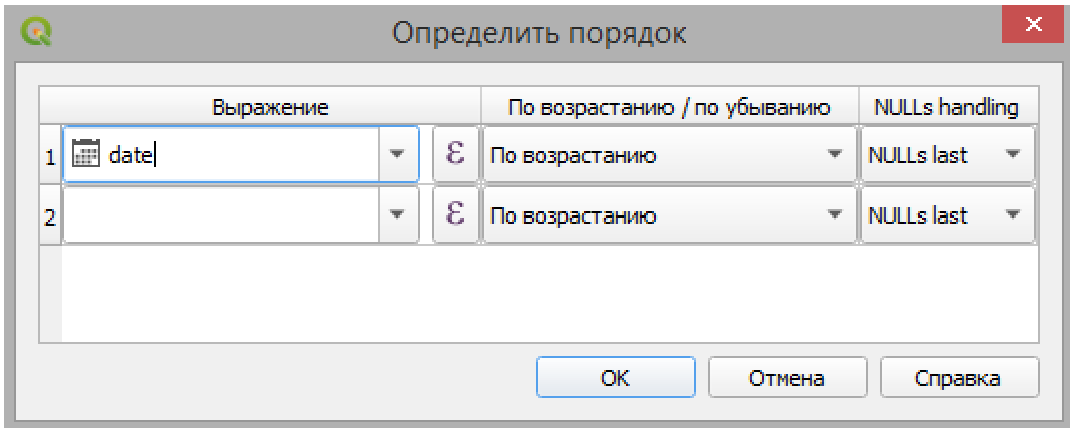


Figure 51 – Parameters of layer rendering

*Opacity* – this property is set for all layers of symbology (parts of combined symbol).

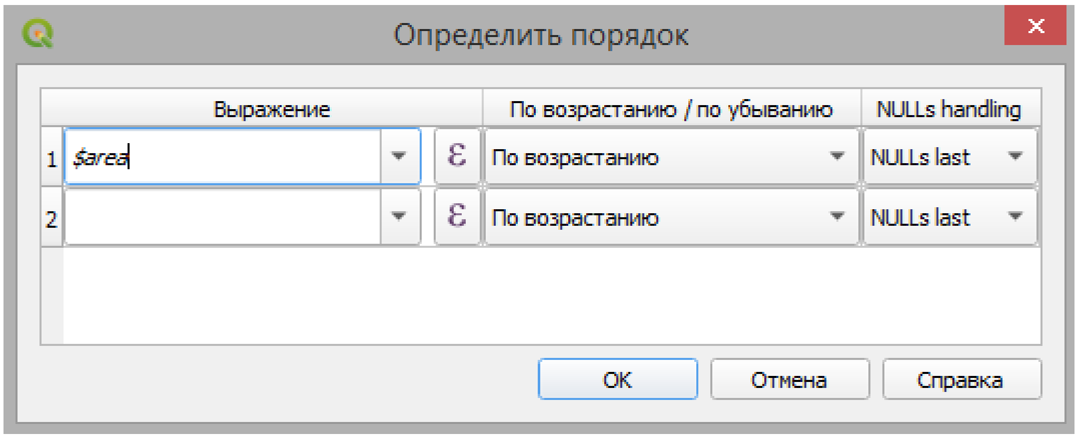
*Order of features rendering* – this property allows you to set sorting of the layer features by specific field or expression, see Figure 52.



| | Выражение | По возрастанию / по убыванию | NULLs handling |
|---|---|---|---|
| 1 | date | По возрастанию | NULLs last |
| 2 | | По возрастанию | NULLs last |

Figure 52 – Order of sorting layer features

The sorting of features is possible for number value, text, date. Also variables *$area* for polygon layer and *$length* for polyline layer are supported, in order to sort the rendering of the layer features by area/length, respectively, see Figure 53.



| | Выражение | По возрастанию / по убыванию | NULLs handling |
|---|---|---|---|
| 1 | $area | По возрастанию | NULLs last |
| 2 | | По возрастанию | NULLs last |

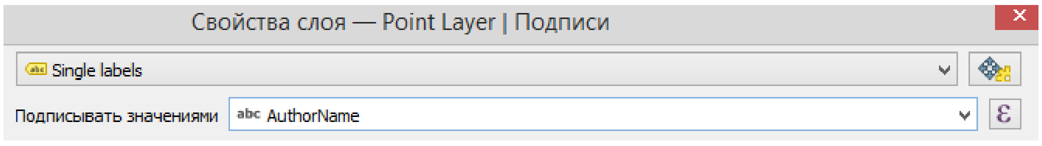
Figure 53 – Example of sorting features by area

## 5.3. Labels

### 5.3.1. Methods of specifying labels

**eLiteGIS** supports the following methods of specifying labels set at the level of map project in QGIS:

- No labels;
- Single label – labeling by values from selected field, see Figure 54;

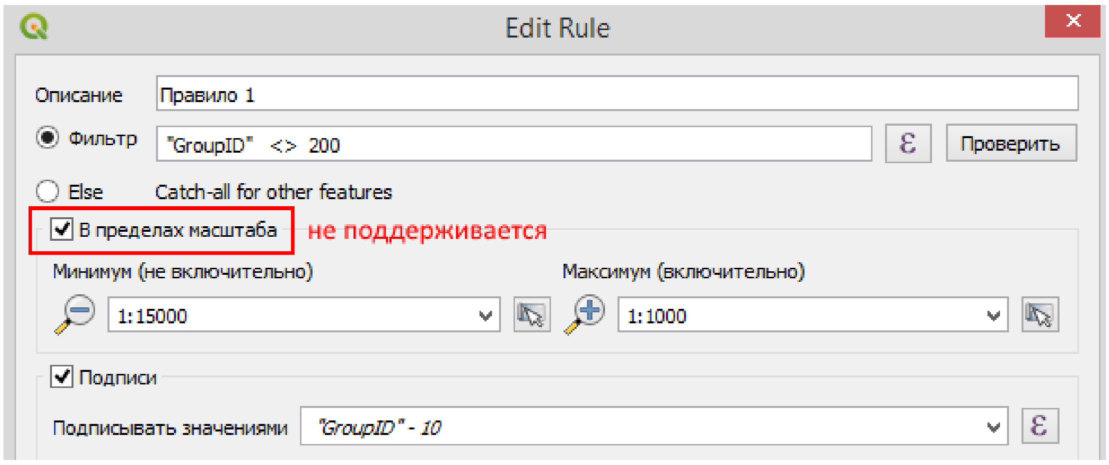

Свойства слоя — Point Layer | Подписи

Single labels

Подписывать значениями: AuthorName

Figure 54 – Single label

- Labels based on rules – label visibility is specified based on SQL filter and label text – based on SQL expression, see Figure 55.


Edit Rule

Описание: Правило 1

Фильтр: "GroupID" <> 200

Else: Catch-all for other features

☑ В пределах масштаба — не поддерживается

Минимум (не включительно): 1:15000

Максимум (включительно): 1:1000

☑ Подписи

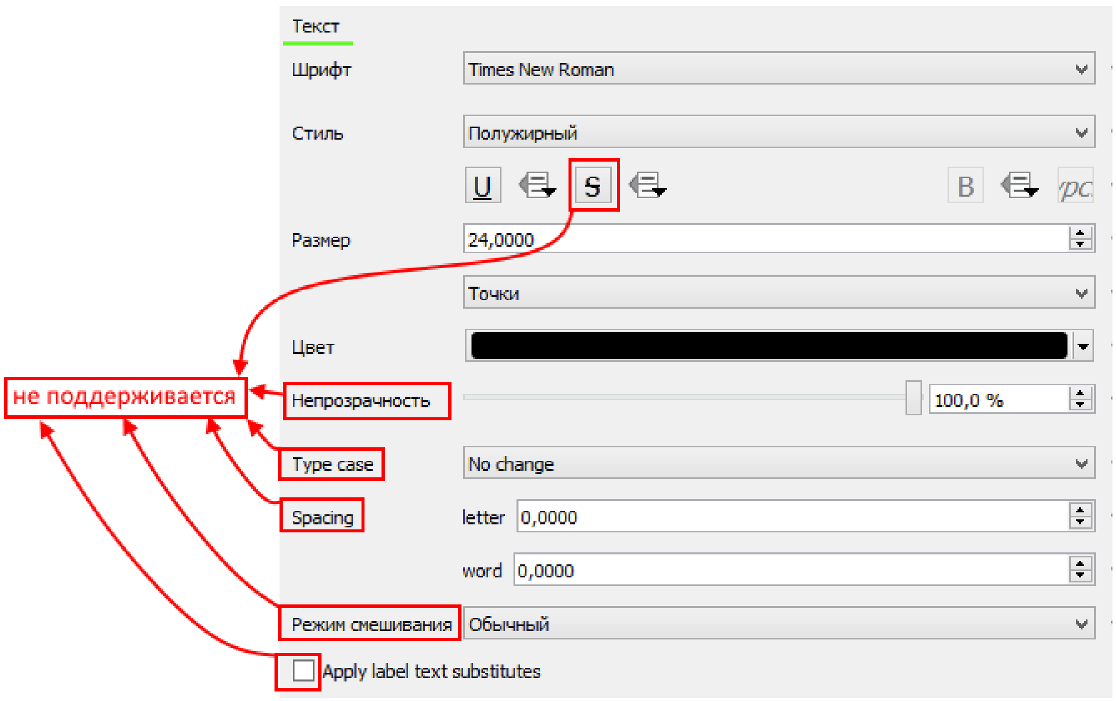
Подписывать значениями: "GroupID" - 10

Figure 55 – Labels based on rules

### 5.3.2. Settings

**eLiteGIS** supports the following labels settings specified at the level of map project in QGIS:

- Text, see Figure 56;


Текст

Шрифт: Times New Roman

Стиль: Полужирный

Размер: 24,0000

Точки

Цвет

Непрозрачность: 100,0 %

Type case: No change

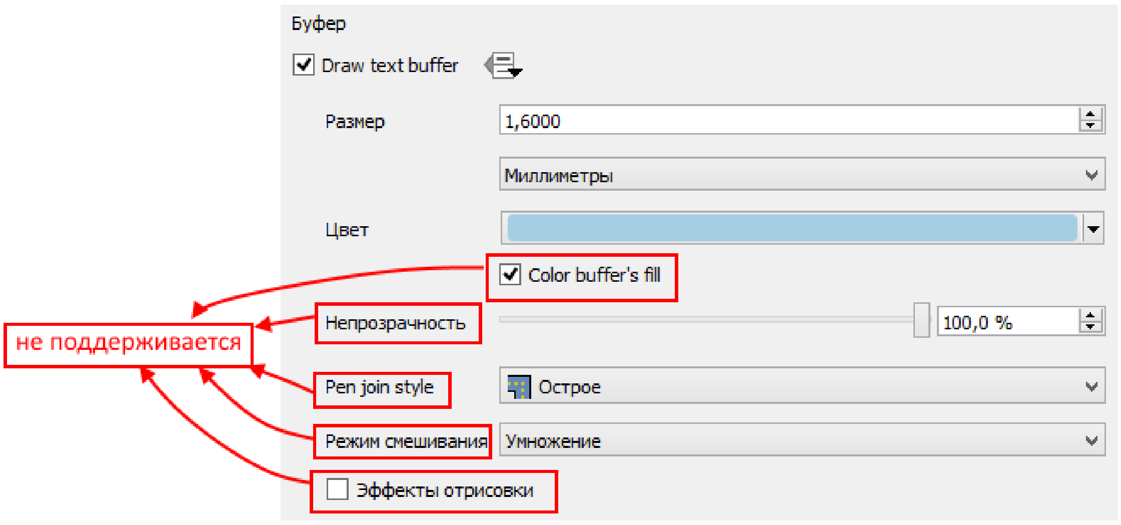
Spacing: letter 0,0000; word 0,0000

Режим смешивания: Обычный

☐ Apply label text substitutes

Figure 56 – Settings for label text

- Buffer, see Figure 57;


Буфер

☑ Draw text buffer

Размер: 1,6000

Миллиметры

Цвет

☑ Color buffer's fill

Непрозрачность: 100,0 %

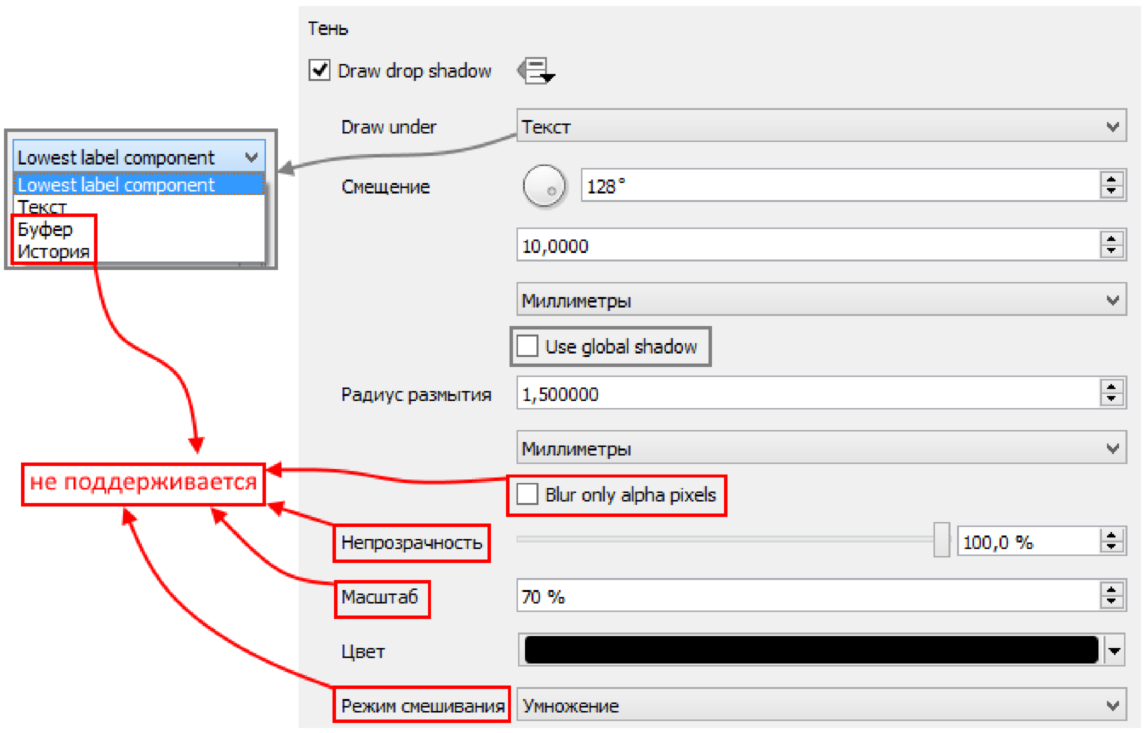
Pen join style: Острое

Режим смешивания: Умножение

☐ Эффекты отрисовки

Figure 57 – Settings for label buffer

- Shadow, see Figure 58;


Тень

[✓] Draw drop shadow

Draw under: Текст

Lowest label component / Lowest label component / Текст / Буфер / История

Смещение: 128°

10,0000

Миллиметры

[ ] Use global shadow

Радиус размытия: 1,500000

Миллиметры

[ ] Blur only alpha pixels

Непрозрачность: 100,0 %

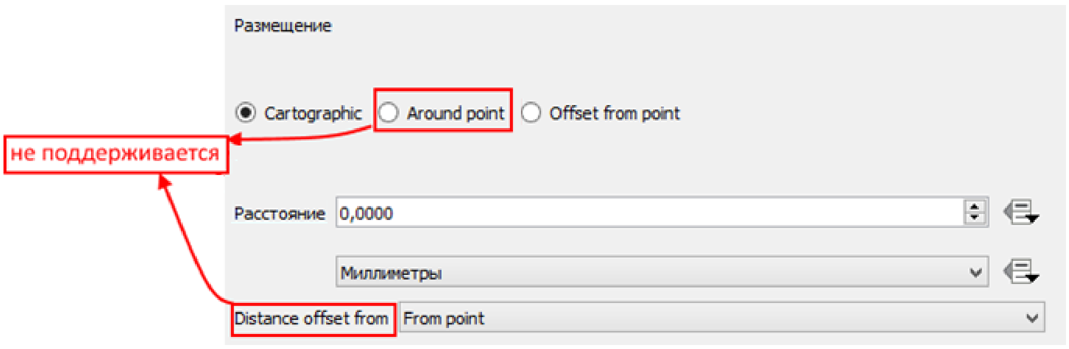
Масштаб: 70 %

Цвет: ■

Режим смешивания: Умножение

Figure 58 – Settings for label shadow

- Location for point feature – only *Cartographic* option is supported by default, see Figure 59.


Размещение

(●) Cartographic ( ) Around point ( ) Offset from point

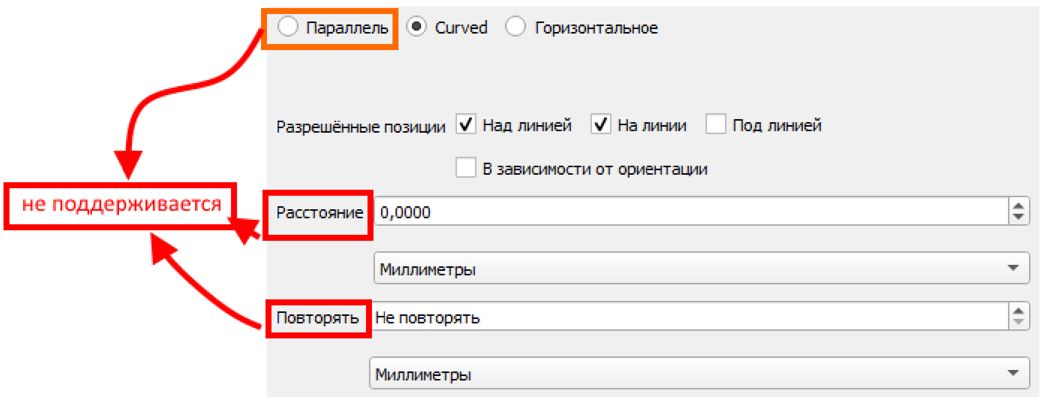
Расстояние: 0,0000

Миллиметры

Distance offset from: From point

Figure 59 – Settings for location of point feature labels

- Location for linear feature – only *Curved* and *Horizontal* options are supported. *Parallel* is shown as *Curved*, see Figure 60;


( ) Параллель (●) Curved ( ) Горизонтальное

Разрешённые позиции: [✓] Над линией [✓] На линии [ ] Под линией

[ ] В зависимости от ориентации

Расстояние: 0,0000

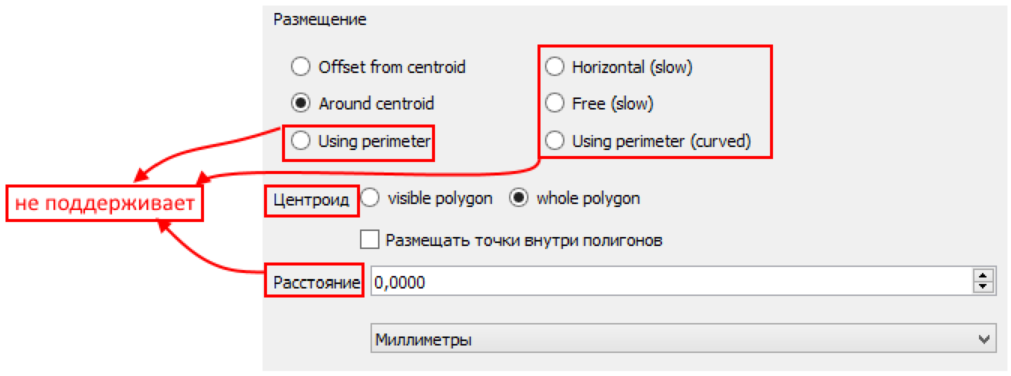
Миллиметры

Повторять: Не повторять

Миллиметры

Figure 60 – Settings for location of linear feature labels

- Location for areal feature – only location in the feature centroid is supported by default, see Figure 61;



Размещение

( ) Offset from centroid
(●) Around centroid
( ) Using perimeter
( ) Horizontal (slow)
( ) Free (slow)
( ) Using perimeter (curved)

не поддерживает

Центроид ( ) visible polygon (●) whole polygon

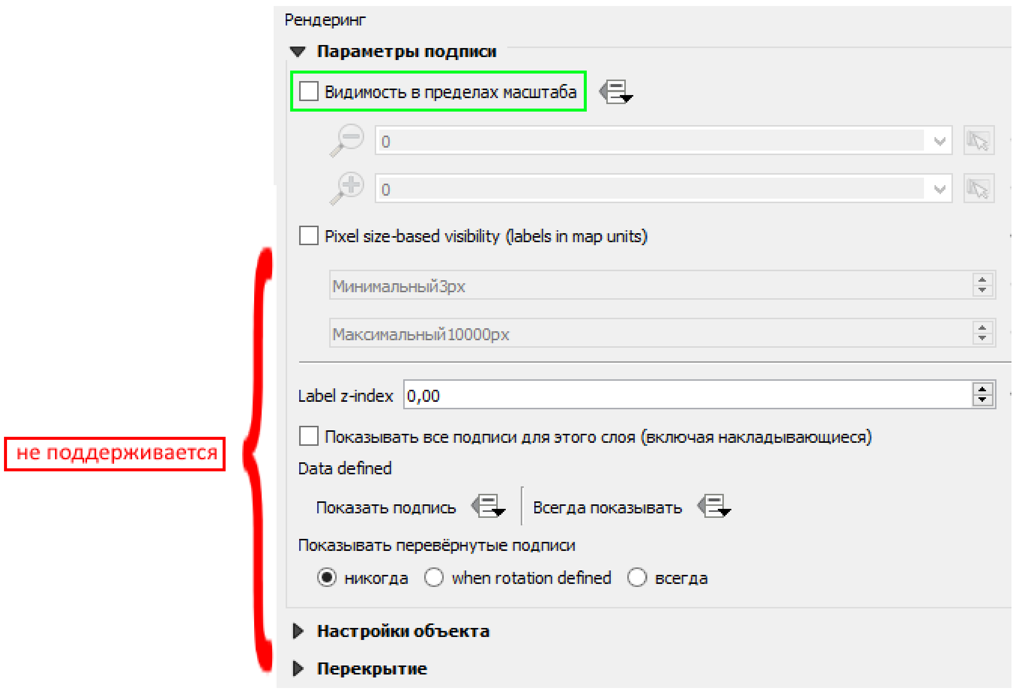
☐ Размещать точки внутри полигонов

Расстояние 0,0000

Миллиметры

Figure 61 – Settings for location of areal feature labels

- Rendering – only visibility within scale is supported, see Figure 62;



Рендеринг

▼ **Параметры подписи**

☐ Видимость в пределах масштаба

0

0

☐ Pixel size-based visibility (labels in map units)

Минимальный3px

Максимальный10000px

Label z-index 0,00

☐ Показывать все подписи для этого слоя (включая накладывающиеся)

Data defined

Показать подпись | Всегда показывать

Показывать перевёрнутые подписи

(●) никогда ( ) when rotation defined ( ) всегда

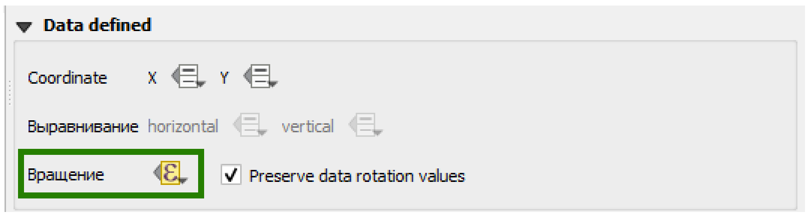
▶ **Настройки объекта**

▶ **Перекрытие**

не поддерживается

Figure 62 – Settings for label rendering

- Rotation – supported for annotations only, see section 6.7 and Figure 63.



▼ **Data defined**

Coordinate X Y

Выравнивание horizontal vertical

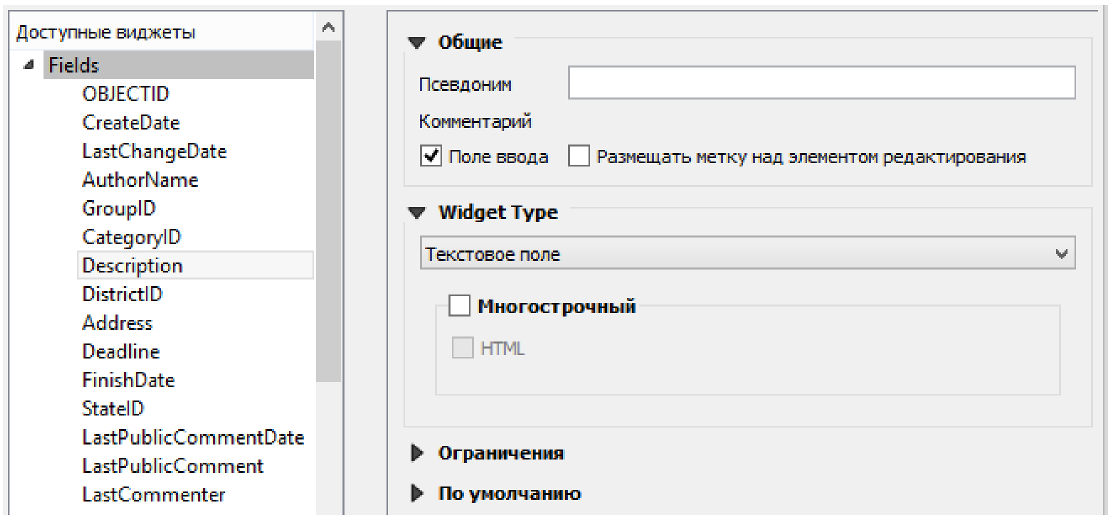
Вращение ☑ Preserve data rotation values

Figure 63 – Settings for rotation

Formatting and history settings are not supported.

## 5.4. Properties of layer fields (Attributes Form)

**eLiteGIS** supports different properties of the layer fields specified at the level of map project, see Figure 64.


Доступные виджеты
- Fields
  - OBJECTID
  - CreateDate
  - LastChangeDate
  - AuthorName
  - GroupID
  - CategoryID
  - Description
  - DistrictID
  - Address
  - Deadline
  - FinishDate
  - StateID
  - LastPublicCommentDate
  - LastPublicComment
  - LastCommenter

Общие
Псевдоним
Комментарий
[✓] Поле ввода ☐ Размещать метку над элементом редактирования

Widget Type
Текстовое поле
☐ Многострочный
☐ HTML

Ограничения
По умолчанию

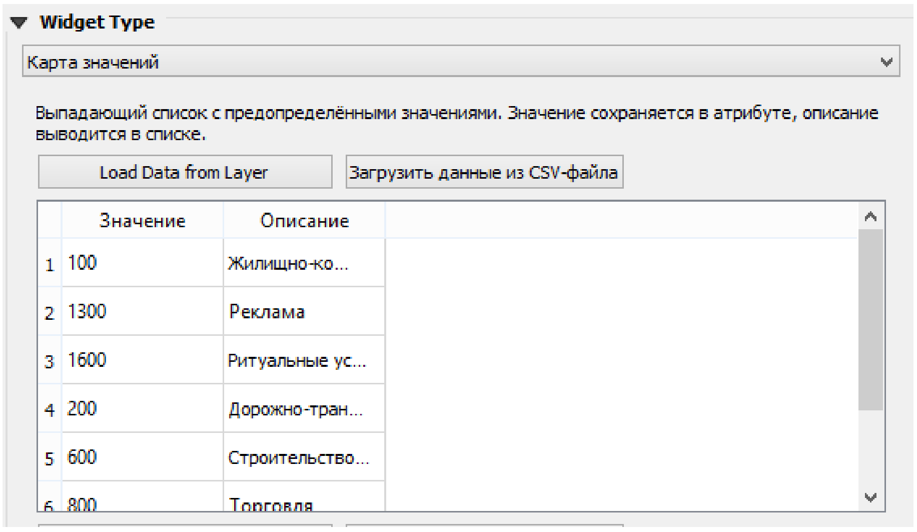
Figure 64 – Setting properties of layer fields

For example, the following properties can be specified:

- *General* – setting the alias.
- *Field type (Widget Type)* – general settings of field type.

*Note: the field type is set by default based on the field type in the database.*

- Domains, see Figure 65.


Widget Type
Карта значений

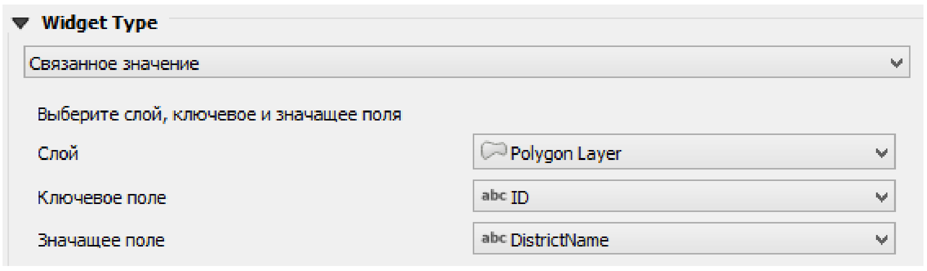
Выпадающий список с предопределёнными значениями. Значение сохраняется в атрибуте, описание выводится в списке.

Load Data from Layer | Загрузить данные из CSV-файла

| | Значение | Описание |
|---|---|---|
| 1 | 100 | Жилищно-ко... |
| 2 | 1300 | Реклама |
| 3 | 1600 | Ритуальные ус... |
| 4 | 200 | Дорожно-тран... |
| 5 | 600 | Строительство... |
| 6 | 800 | Торговля |

Figure 65 – Setting domain for the field

- Related value – domain values based on values from another layer, see Figure 66.

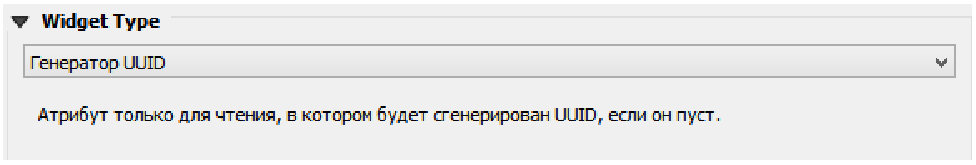

Widget Type
Связанное значение

Выберите слой, ключевое и значащее поля

| | |
|---|---|
| Слой | Polygon Layer |
| Ключевое поле | abc ID |
| Значащее поле | abc DistrictName |

Figure 66 – Setting related values for the field

- UUID generator - generation of Universally unique identifier by feature’s creation, see Figure 67. The required field type is *uuid* or *varchar*.



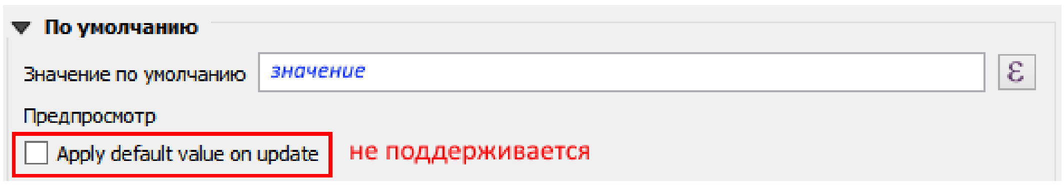
**Widget Type**

Генератор UUID

Атрибут только для чтения, в котором будет сгенерирован UUID, если он пуст.

Figure 67 – Setting UUID generator

- *Default values (Default expression)* – values that will be recorded to the appropriate fields by creation of the feature, see Figure 68.



**По умолчанию**

Значение по умолчанию: значение

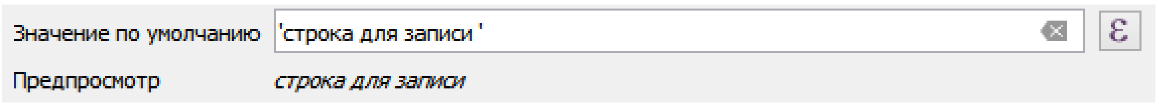
Предпросмотр

☐ Apply default value on update — не поддерживается

Figure 68 – Setting default values for the field

Types of values:

- String. Field type – string, record of specified string, see Figure 69.

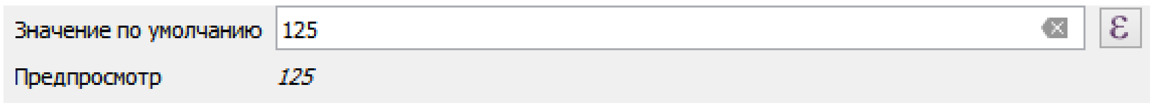


Значение по умолчанию: 'строка для записи '

Предпросмотр: строка для записи

Figure 69 – Default value for the string

- Integer. Field type – integer, record of specified integer, see Figure 70.

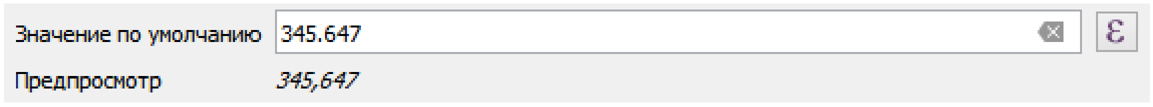


Значение по умолчанию: 125

Предпросмотр: 125

Figure 70 – Default value for the integer

- Real number. Field type – real, record of specified real number, see Figure 71.

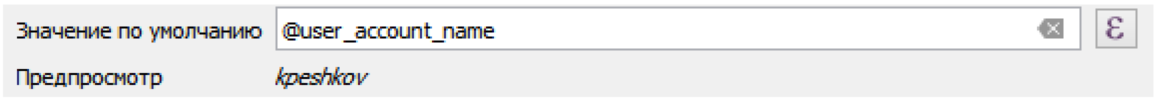


Значение по умолчанию: 345.647

Предпросмотр: 345,647

Figure 71 – Default value for the real number

- @user_account_name – record of the current user name in **eLiteGIS** {CurrentUser}. If the user is not authorized, *null* will be recorded. Field type - string, see Figure 72.

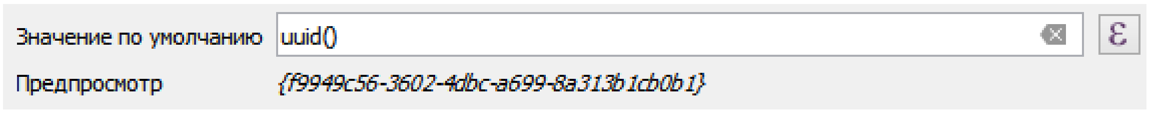


Значение по умолчанию: @user_account_name

Предпросмотр: kpeshkov

Figure 72 – Current user name used as the default value

- uuid() – record of the *Universally unique identifier*. Field type is *uuid* or string, see Figure 73.

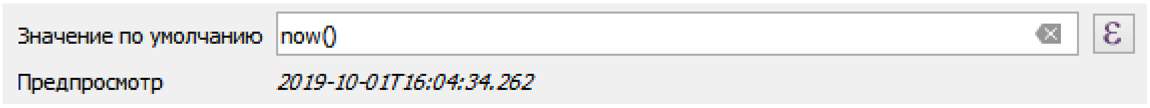


Значение по умолчанию uuid()

Предпросмотр {f9949c56-3602-4dbc-a699-8a313b1cb0b1}

Figure 73 – Using the generated UUID as the default value

- now() – record of the current date-time UTC, {CurrentDateTime}. Field type - *timestamptz*, see Figure 74.

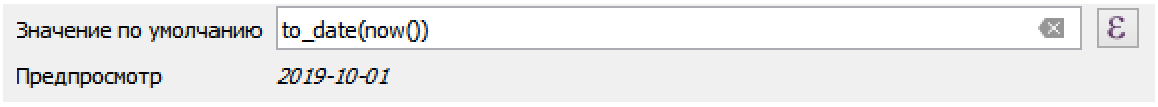


Значение по умолчанию now()

Предпросмотр 2019-10-01T16:04:34.262

Figure 74 – Using the current date-time as the default value

- to_date(now()) – record of the current date in local time zone, {CurrentDate}. Field type – *date*, see Figure 75.



Значение по умолчанию to_date(now())

Предпросмотр 2019-10-01

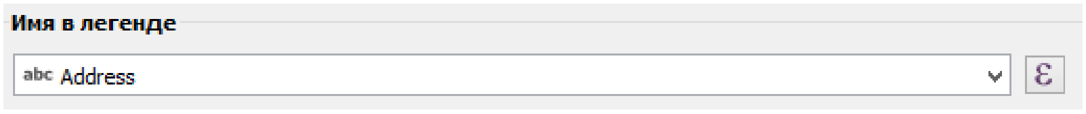
Figure 75 – Using the current date and time in local time zone as the default value

## 5.5.Joins

Joins (*Join Field*) specified at the level of map project in QGIS are not supported at **eLiteGIS** level by publishing project as map service.

## 5.6.Displaying value by feature's identification

**eLiteGIS** supports value specified at QGIS project level, which value is used for display of the feature's identification result, for example, on the map in **CoGIS**, see Figure 76.



**Имя в легенде**

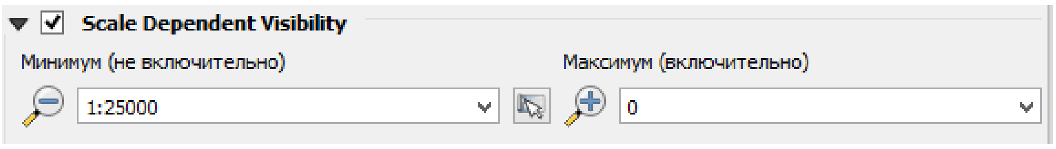
abc Address

Figure 76 – Field values by feature's identification

SQL expressions are partly supported, see section 7.2 for details.

## 5.7.Layer visibility settings

**eLiteGIS** supports settings of layer visibility in the specific scale range or in all scales, set in the map project, see Figure 77.



☑ **Scale Dependent Visibility**

Минимум (не включительно) 1:25000

Максимум (включительно) 0

Figure 77 – Visibility settings via layer parameters

Setting/clearing the layer visibility scale ranges can be done in the context menu of the layer, see Figure 78.

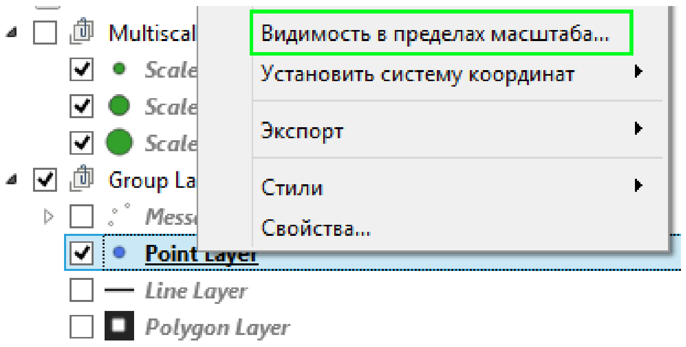
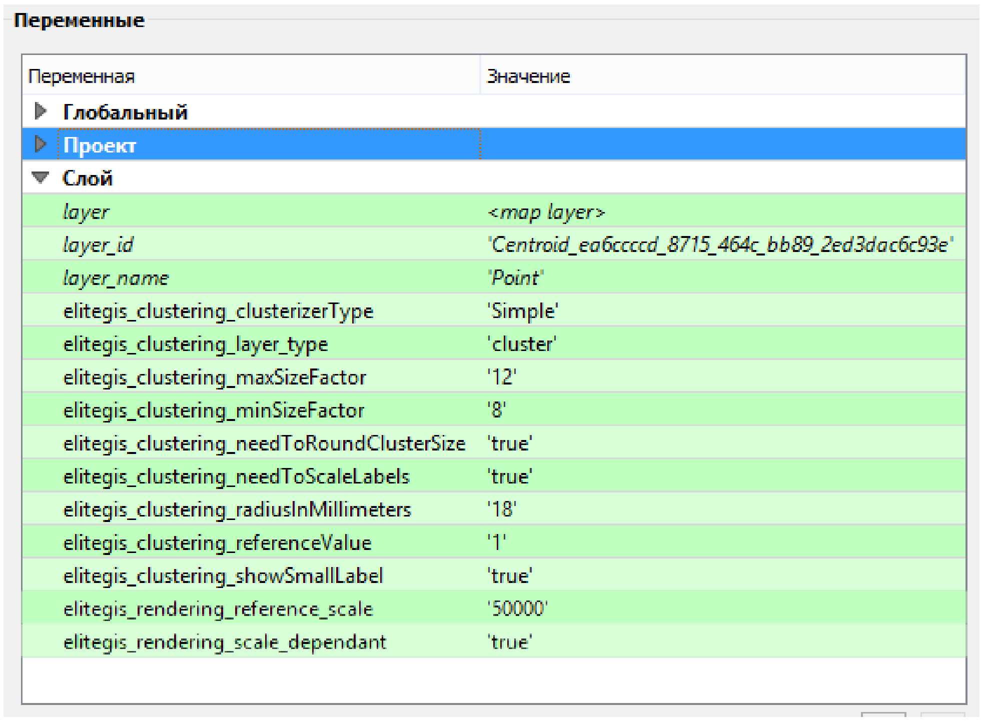


Figure 78 – Setting the layer visibility via context menu

## 5.8. Variables

At the level of single layer in QGIS project it is possible to set variables that will be used by publishing services in **eLiteGIS**, for example: clusterization settings (*Clustering*), scale dependency of symbology (*Scale Dependent*), tracking edits (*Edit Tracker*) and more (see section 6), see Figure 79.



**Переменные**

| Переменная | Значение |
|---|---|
| **Глобальный** | |
| **Проект** | |
| **Слой** | |
| *layer* | *<map layer>* |
| *layer_id* | *'Centroid_ea6ccccd_8715_464c_bb89_2ed3dac6c93e'* |
| *layer_name* | *'Point'* |
| elitegis_clustering_clusterizerType | 'Simple' |
| elitegis_clustering_layer_type | 'cluster' |
| elitegis_clustering_maxSizeFactor | '12' |
| elitegis_clustering_minSizeFactor | '8' |
| elitegis_clustering_needToRoundClusterSize | 'true' |
| elitegis_clustering_needToScaleLabels | 'true' |
| elitegis_clustering_radiusInMillimeters | '18' |
| elitegis_clustering_referenceValue | '1' |
| elitegis_clustering_showSmallLabel | 'true' |
| elitegis_rendering_reference_scale | '50000' |
| elitegis_rendering_scale_dependant | 'true' |

Figure 79 – Setting variables for the layer

## 6. Variables settings

In QGIS it is possible to set variables both at the level of the entire project and for the separate layers.

These variables can be further used by publishing services in **eLiteGIS**. These variables provide the extended options for data visualization (heat maps, clusterization, pie charts etc.) and for data management (relationships between features of different layers, tracking edits, reference books of values etc.).

No additional settings at the **eLiteGIS** or **CoGIS** levels are required.

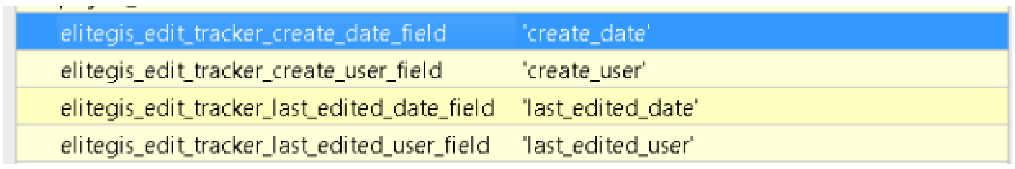
### 6.1. Tracking edits (Edit Tracker)

The settings group *Edit Tracker* includes settings for automatic completion of fields with values, see Table 1.

Table 1 – Edit Tracker settings group

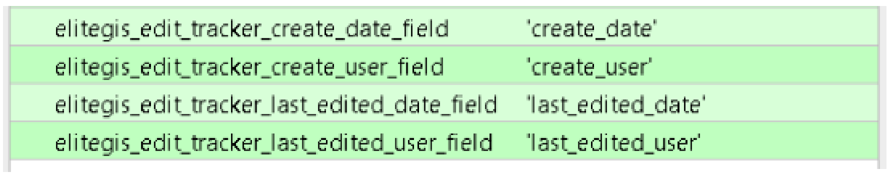
| Variable | Description | Properties for |
|---|---|---|
| *eLiteGIS_edit_tracker_create_date_field* | *Field name for completing the date of the project's creation* | *Entire project and layer* |
| *eLiteGIS_edit_tracker_create_user_field* | *Field name for completing the name of the user who created the feature* | *Entire project and layer* |
| *eLiteGIS_edit_tracker_last_edited_date_field* | *Field name for completing the date of the feature's editing* | *Entire project and layer* |
| *eLiteGIS_edit_tracker_last_edited_user_field* | *Field name for completing the name of the user who edited the feature* | *Entire project and layer* |

Below are examples of settings specified for the entire project (Figure 80) and for the layer (Figure 81).



| | |
|---|---|
| elitegis_edit_tracker_create_date_field | 'create_date' |
| elitegis_edit_tracker_create_user_field | 'create_user' |
| elitegis_edit_tracker_last_edited_date_field | 'last_edited_date' |
| elitegis_edit_tracker_last_edited_user_field | 'last_edited_user' |

Figure 80 – Setting Edit Tracker variables for the project



| | |
|---|---|
| elitegis_edit_tracker_create_date_field | 'create_date' |
| elitegis_edit_tracker_create_user_field | 'create_user' |
| elitegis_edit_tracker_last_edited_date_field | 'last_edited_date' |
| elitegis_edit_tracker_last_edited_user_field | 'last_edited_user' |

Figure 81 – Setting Edit Tracker variables for the layer

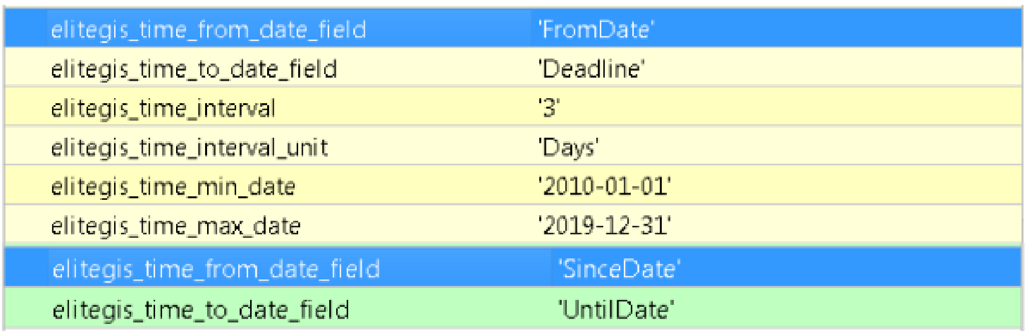
### 6.2. Time properties (Time Extent)

The settings group *Time Extent* includes settings for time properties for visualization of data using *Time Slider*, see Table 2.

Table 2 –Time Extent settings group

| Variable | Description | Properties for |
|---|---|---|
| *eLiteGIS_time_from_date_field* | *Field name with starting value of the time range* | *Entire project and layer* |
| *eLiteGIS_time_to_date_field* | *Field name with ending value of the time range* | *Entire project and layer* |
| *eLiteGIS_time_interval* | *Value of the time range* | *Entire project* |
| *eLiteGIS_time_interval_unit* | *Units of the value of the time range* | *Entire project* |
| *eLiteGIS_time_min_date* | *Value of the minimum date of the time range* | *Entire project* |
| *eLiteGIS_time_max_date* | *Value of the maximum date of the time range* | *Entire project* |
| *eLiteGIS_time_time_data_cumulative* | *No value. Includes cumulative effect.* | *Layer* |

Below are examples of settings specified for the entire project and for the layer, see Figure 82.



| | |
|---|---|
| elitegis_time_from_date_field | 'FromDate' |
| elitegis_time_to_date_field | 'Deadline' |
| elitegis_time_interval | '3' |
| elitegis_time_interval_unit | 'Days' |
| elitegis_time_min_date | '2010-01-01' |
| elitegis_time_max_date | '2019-12-31' |
| elitegis_time_from_date_field | 'SinceDate' |
| elitegis_time_to_date_field | 'UntilDate' |

Figure 82 – Setting Time Extent variables for the project and for the layer

Variable elitegis_time_time_data_cumulative '' includes the cumulative effect when not the time range or specific date is selected, but the right limit of the time range that is known to include all the history to the selected date.

## 6.3.Clusterization

At the map project level it is possible to set clusterization of features (grouping features and displaying groups in the group's centroid) that will be supported by publishing project in **eLiteGIS** and by adding this service to the map in **CoGIS**.

### 6.3.1. Cluster symbol

The cluster coloring is set according to the rules of the symbology settings. The cluster marker can be simple, symbol, raster, SVG and combined, see Figure 83.

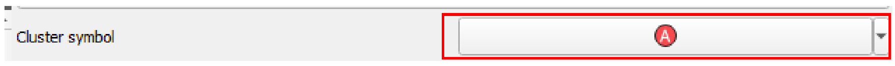


Figure 83 – Selecting marker for cluster

The cluster symbol size is set via the Data defined override expression, see Figure 84.

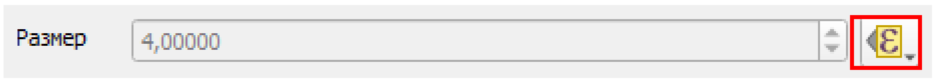
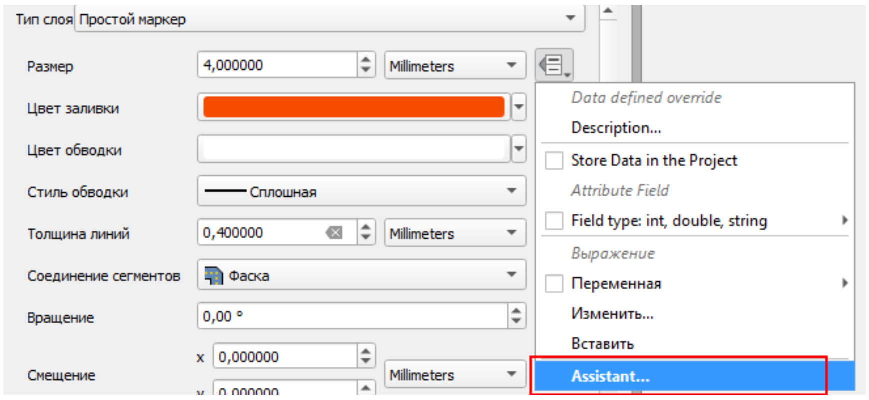


Figure 84 – Setting the cluster symbol size (1)

To do so, in the context menu of Data defined override expression select Assistant, see Figure 85.


Тип слоя Простой маркер

Размер 4,000000 Millimeters

Цвет заливки

Цвет обводки

Стиль обводки Сплошная

Толщина линий 0,400000 Millimeters

Соединение сегментов Фаска

Вращение 0,00 °

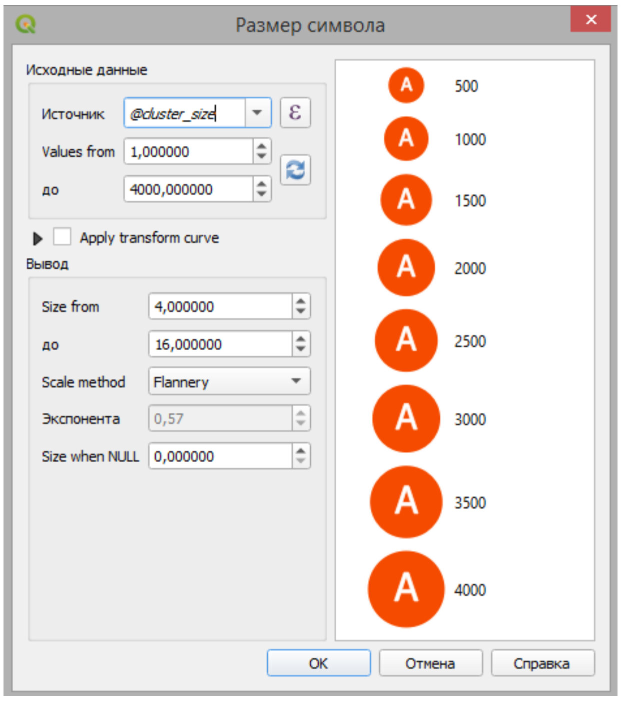
Смещение x 0,000000 y 0,000000 Millimeters

Figure 85 – Setting the cluster symbol size (2)

In the Symbol size window, the size of the symbol is specified.

The source of values for the size calculation is the *@cluster_size* variable that contains the number of features appeared in the cluster.

The fields *Values from* and *To* show the number of features that can appear to the cluster, see Figure 86.


Размер символа

Исходные данные

Источник @cluster_size

Values from 1,000000

до 4000,000000

Apply transform curve

Вывод

Size from 4,000000

до 16,000000

Scale method Flannery

Экспонента 0,57

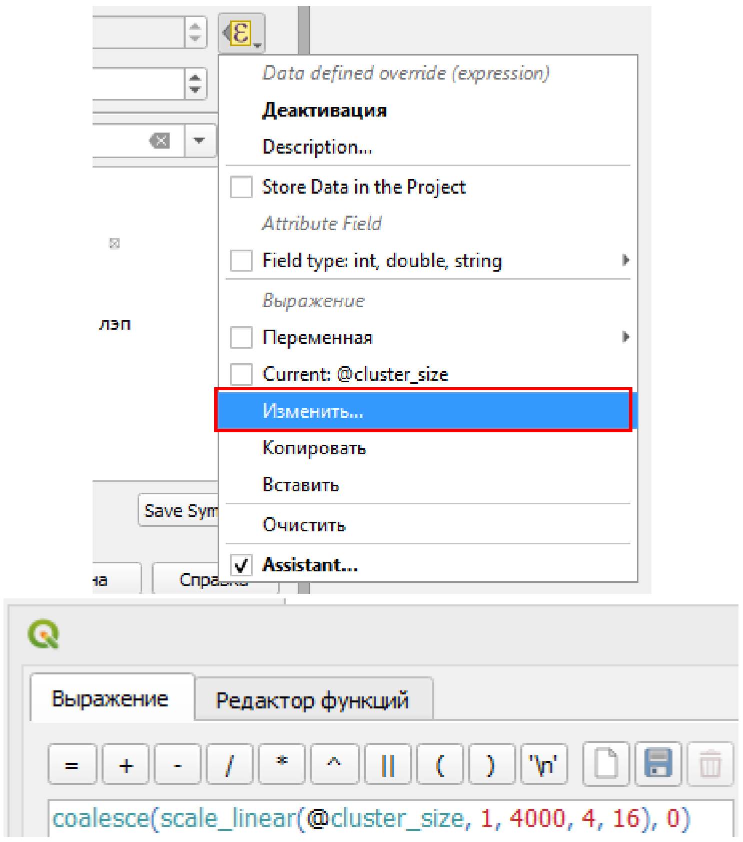
Size when NULL 0,000000

Figure 86 – Setting the cluster symbol size (3)

That is, the *Assistant* tool allows you to create the formula for calculation of the cluster symbol size as following:

```
coalesce(scale_linear(@cluster_size, 1, 4000, 4, 16), 0)
```

The size calculation formula can be edited or set without the *Assistant* tool, to do so, select *Edit* in the *Data defined override expression* menu, see Figure 87.



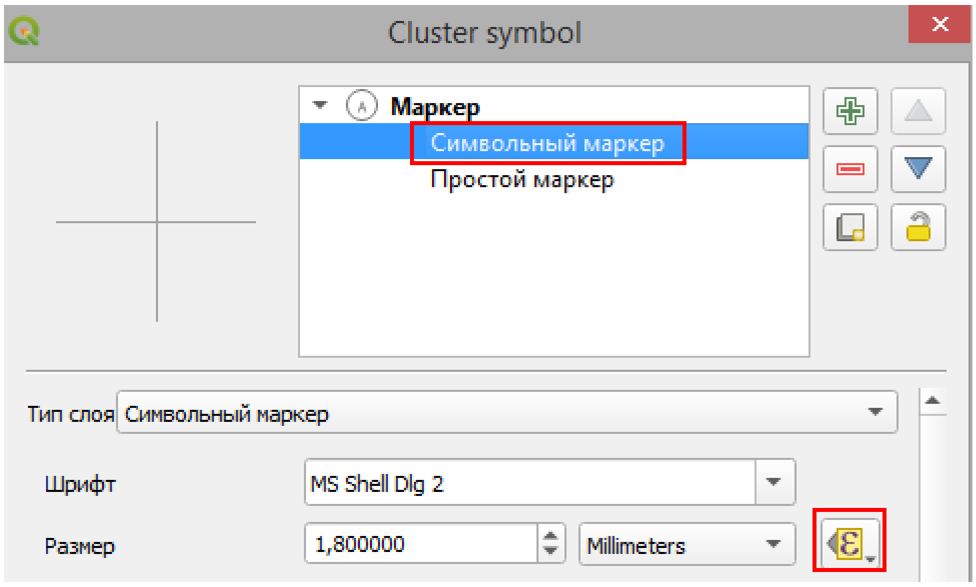
coalesce(scale_linear(@cluster_size, 1, 4000, 4, 16), 0)

Figure 87 – Setting the cluster symbol size (4)

### 6.3.2. Cluster label

The label of number of features appeared in the cluster is set in the cluster symbol coloring marker, Symbol marker layer, see Figure 88

*Note: for the Point cluster symbology, the Symbol marker is provided by default.*



Тип слоя: Символьный маркер
Шрифт: MS Shell Dlg 2
Размер: 1,800000 Millimeters

Figure 88 – Setting the label of number of features appeared in the cluster

The size of symbol marker is set by default same as the size of the entire symbol for cluster, using the same formula, but with the added coefficient:

```
0.5*(coalesce(scale_linear(@cluster_size, 1, 4000, 4, 16), 0))
```

For additional setting of the label, the following parameters are used, see Table 3.

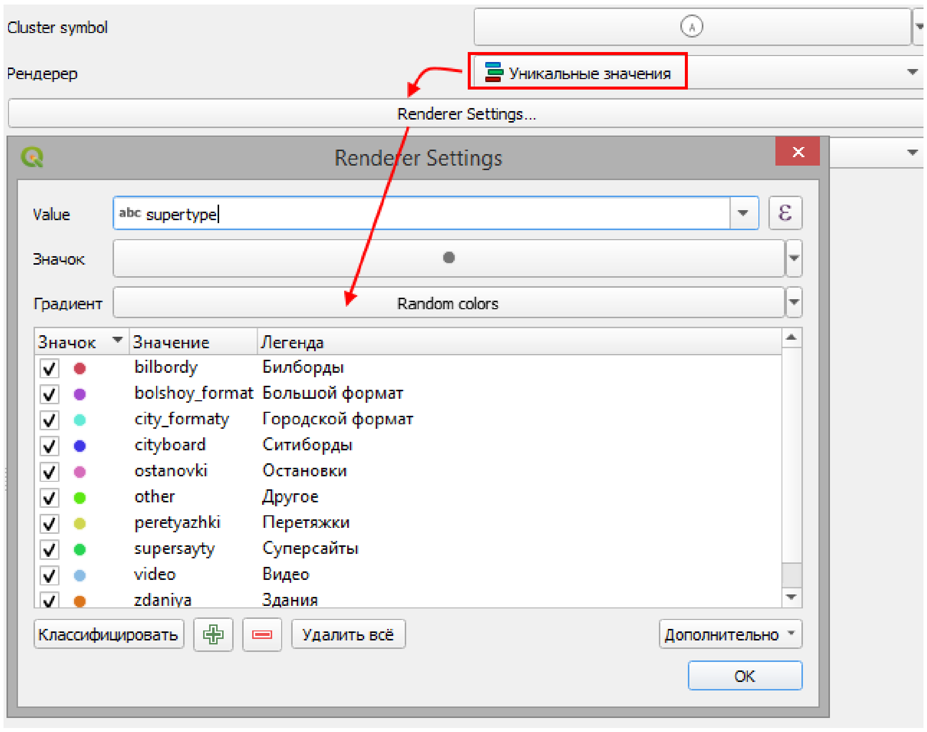
Table 3 – Parameters for additional setting of label

| Variable | Description | Value |
|---|---|---|
| *eLiteGIS_clustering_needToScaleLabels* | *Scaling labels* | *true/false* |
| *eLiteGIS_clustering_showSmallLabel* | *Rounding the number of features in the cluster (not more than 4 characters)* | *true/false* |

### 6.3.3. Rendering the source features (Renderer Settings)

The source features can be rendered using the following symbology types: *No symbols, Simple symbol, Unique values, Graduated symbol, Rules.*

In the *Renderer Settings* the coloring for each type of symbology is set, same as for simple not cluster features (see section 5.2.9), see Figure 89.



Cluster symbol

Рендерер: Уникальные значения

Renderer Settings...

Renderer Settings

Value: supertype

Значок: ●

Градиент: Random colors

| Значок | Значение | Легенда |
|---|---|---|
| ✓ ● | bilbordy | Билборды |
| ✓ ● | bolshoy_format | Большой формат |
| ✓ ● | city_formaty | Городской формат |
| ✓ ● | cityboard | Ситиборды |
| ✓ ● | ostanovki | Остановки |
| ✓ ● | other | Другое |
| ✓ ● | peretyazhki | Перетяжки |
| ✓ ● | supersayty | Суперсайты |
| ✓ ● | video | Видео |
| ✓ ● | zdaniya | Здания |

Figure 89 – Rendering the source features of the cluster

### 6.3.4. Distance

This group of settings is provided for setting the coverage radius for features grouping. Supported measurement units are millimeters.

### 6.3.5. Visibility within scale

The visibility within scale set in QGIS settings affects the cluster visibility. The following additional parameters are used for setting visibility of the source features, see Table 4.

Table 4 – Parameters for setting visibility of the source features of the cluster

| Variable | Description |
|---|---|
| *eLiteGIS_feature_minScale* | *Minimum scale value* |
| *eLiteGIS_feature_maxScale* | *Maximum scale value* |

### 6.3.6. Clusterization type

**eLiteGIS** supports a few algorithms to cluster features. For selection of the required algorithm, the additional parameter is used: `eLiteGIS_clustering_clusterizerType`

This parameter can have the following values: *HexagonedSimple* (by default), *HexagonedDelaunay*, *Simple*, *Delaunay*.

### 6.3.7. Edge symbology

For setting of the edge symbology, the additional layer (temporary layer in QGIS) is used, see Figure 90.

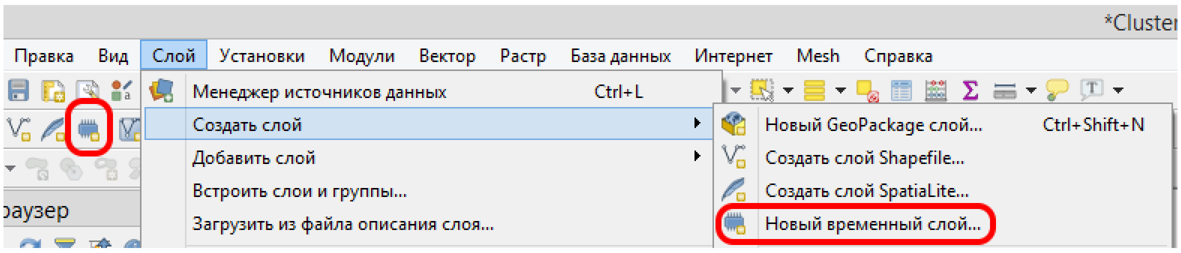

*Cluster
Правка
Вид
Слой
Установки
Модули
Вектор
Растр
База данных
Интернет
Mesh
Справка
Менеджер источников данных Ctrl+L
Создать слой
Добавить слой
Встроить слои и группы...
Загрузить из файла описания слоя...
Новый GeoPackage слой... Ctrl+Shift+N
Создать слой Shapefile...
Создать слой SpatiaLite...
Новый временный слой...
раузер

Figure 90 – New temporary layer for cluster edges

The required geometry type of the temporary layer is *LineString*, see Figure 91.

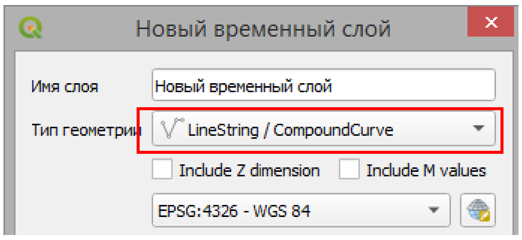


Figure 91 – Geometry type of the temporary layer with cluster edges

The created temporary layer should be placed just under the main cluster layer, see Figure 92.

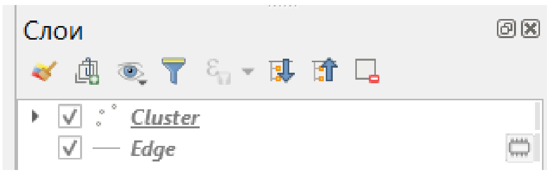


Figure 92 – Location of the layer with cluster edges in the project's layers tree

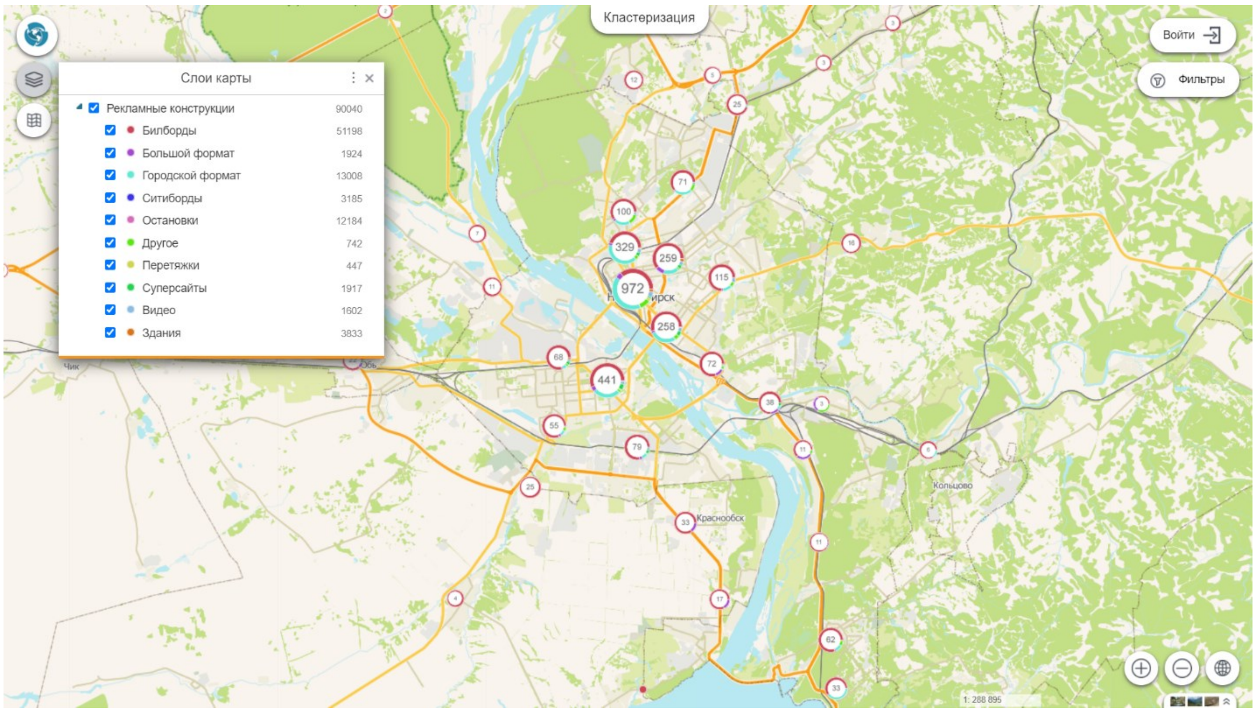
In the layer properties in Variables tab specify parameter `eLiteGIS_clustering_layer_type` with value *edge*:

| | |
|---|---|
| elitegis_clustering_layer_type | 'edge' |

Specify the edge style (coloring). You can set the visibility within the scale. If visibility is not set, this property will be taken from the main cluster layer.

### 6.3.8. Pie charts for cluster

With the help of additional layers and parameters, you can set the display of the features cluster as the pie chart that dynamically changes depending on the types of features included in the cluster. Example of such display is shown on Figure 93.



Слои карты

| | |
|---|---|
| Рекламные конструкции | 90040 |
| ● Билборды | 51198 |
| ● Большой формат | 1924 |
| ● Городской формат | 13008 |
| ● Ситиборды | 3185 |
| ● Остановки | 12184 |
| ● Другое | 742 |
| ● Перетяжки | 447 |
| ● Суперсайты | 1917 |
| ● Видео | 1602 |
| ● Здания | 3833 |

Figure 93 – Clusters of city advertising structures on Novosibirsk map: each cluster is represented as the pie chart on the distribution of structure types in the cluster

For setting of the pie charts the additional permanent layers are used.

For creation of such a setting layer you can duplicate the main cluster layer, see Figure 94.

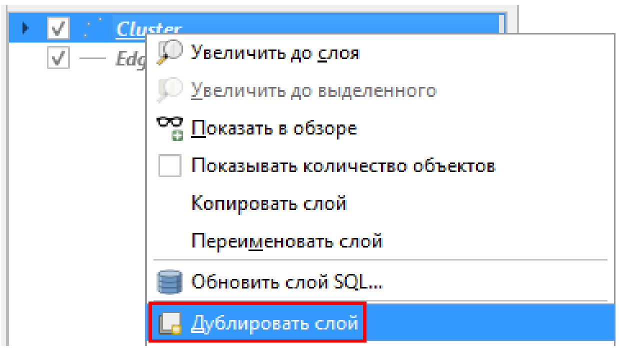


Figure 94 – Duplicating the main cluster layer for creation of the pie chart

With this method of creation, the required data source and the cluster size formula are immediately saved in the setting layer. It is also possible to add such a layer again, recreating all the settings manually. It is necessary to place the setting layer under the main cluster layer. In the case where there is the layer for edges, the location should be as follows, see Figure 95.

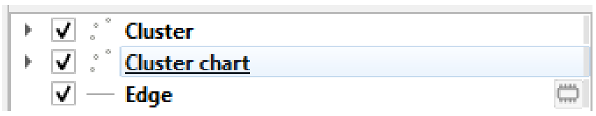


Figure 95 – Location of the setting layer for pie charts relative to the main cluster layer and edges layer

In the properties of the setting layer in the Variables tab specify `eLiteGIS_clustering_layer_type` with value *pie*:

| | |
|---|---|
| elitegis_clustering_layer_type | 'pie' |

.

*Note: Thus, it is indicated that this layer will be perceived by* ***eLiteGIS*** as the setting layer and so will not be published in the service.

The order of drawing symbols in the cluster (there can be many levels) is shown in the figure below, see Figure 96.

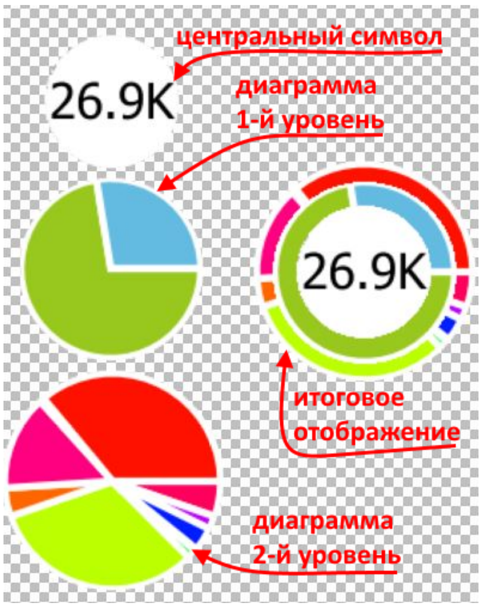


Figure 96 – The order of drawing symbols in the cluster

6.3.8.1. Coloring segments (pies) in the pie chart

For setting the colors of the chart pies, make sure that in the layer properties in Style section the *Point cluster* symbology type is specified.

Then, in *Cluster symbol* set the chart symbol and size.

For symbol, the value Simple marker should be selected (the symbol marker should be deleted in case if it remained by duplicating the source layer).

The size is set by the same formula as in the source layer, with the addition of the coefficient greater by one, so that the pie chart is larger than the central symbol, see Figure 97.

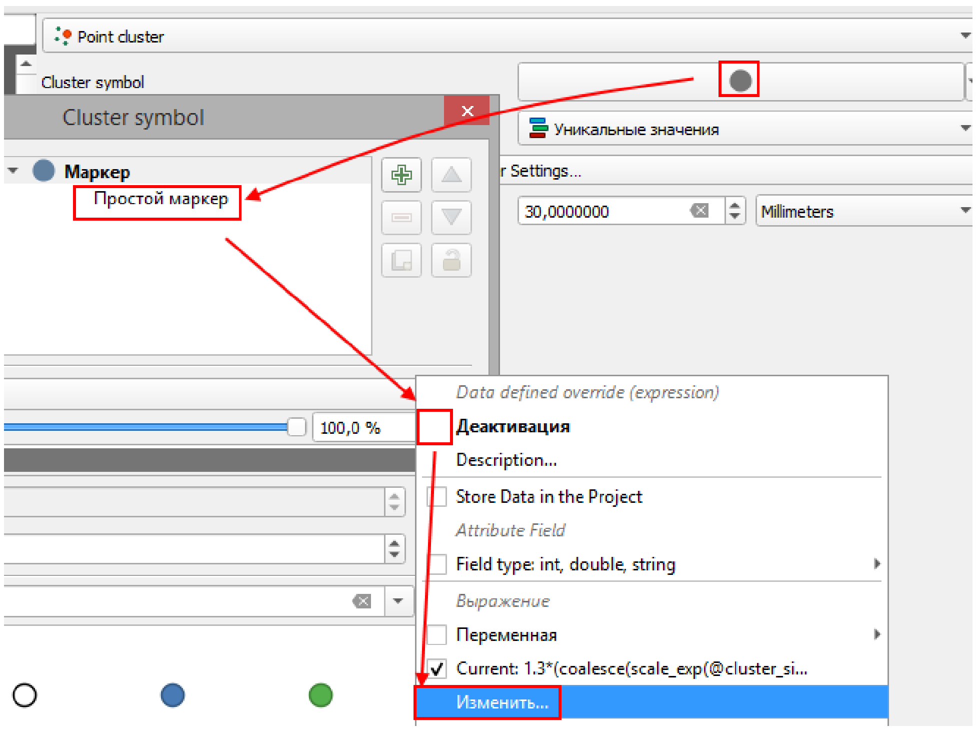
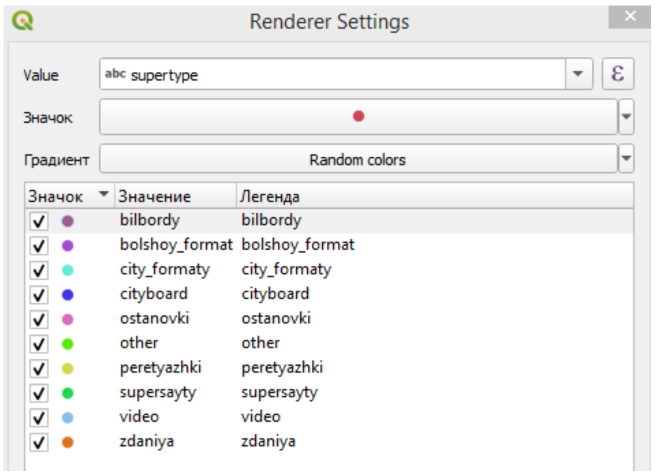


Figure 97 – Setting symbol for the pie chart

Thus, the following formula for calculation of the cluster symbol size is created:

```
1.3*(coalesce(scale_linear(@cluster_size, 1, 4000, 4, 16), 0))
```

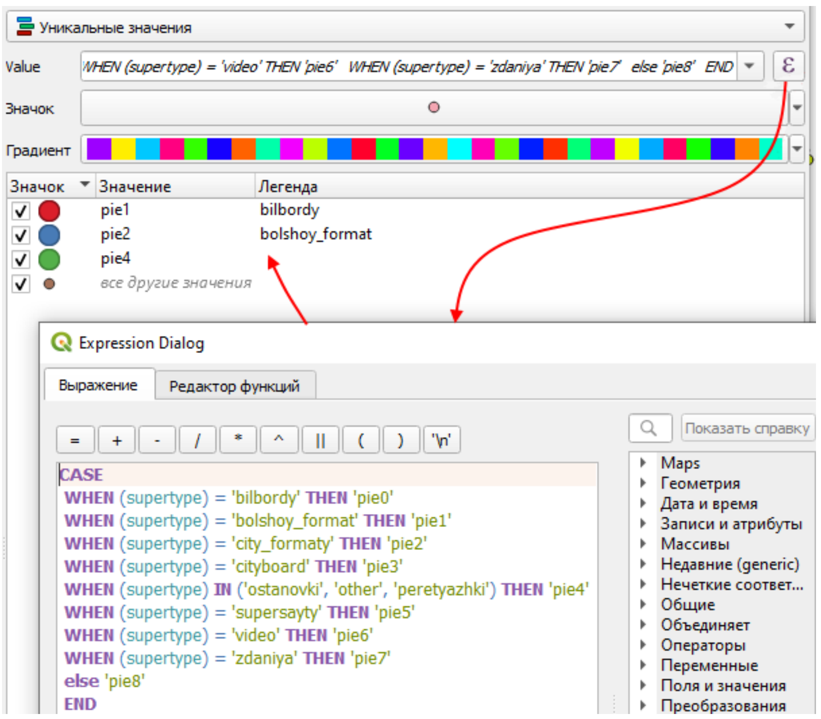
For coloring of the chart pies the rendering type Unique values is used. Each value is given its own symbol, from which the color of the segment is taken for this value in a pie, see Figure 98.



| Значок | Значение | Легенда |
|---|---|---|
| ✓ ● | bilbordy | bilbordy |
| ✓ ● | bolshoy_format | bolshoy_format |
| ✓ ● | city_formaty | city_formaty |
| ✓ ● | cityboard | cityboard |
| ✓ ● | ostanovki | ostanovki |
| ✓ ● | other | other |
| ✓ ● | peretyazhki | peretyazhki |
| ✓ ● | supersayty | supersayty |
| ✓ ● | video | video |
| ✓ ● | zdaniya | zdaniya |

Figure 98 – Setting pie chart colors based on the unique values

For defining values for each segment it is also possible to use SQL expression. For example, this allows you to combine categories, see Figure 99.



Value: WHEN (supertype) = 'video' THEN 'pie6'  WHEN (supertype) = 'zdaniya' THEN 'pie7'  else 'pie8'  END

| Значок | Значение | Легенда |
|---|---|---|
| ✓ ● | pie1 | bilbordy |
| ✓ ● | pie2 | bolshoy_format |
| ✓ ● | pie4 | |
| ✓ ● | все другие значения | |

```
CASE
 WHEN (supertype) = 'bilbordy' THEN 'pie0'
 WHEN (supertype) = 'bolshoy_format' THEN 'pie1'
 WHEN (supertype) = 'city_formaty' THEN 'pie2'
 WHEN (supertype) = 'cityboard' THEN 'pie3'
 WHEN (supertype) IN ('ostanovki', 'other', 'peretyazhki') THEN 'pie4'
 WHEN (supertype) = 'supersayty' THEN 'pie5'
 WHEN (supertype) = 'video' THEN 'pie6'
 WHEN (supertype) = 'zdaniya' THEN 'pie7'
 else 'pie8'
 END
```

Figure 99 – Setting pie chart colors using SQL expression

The visibility within the scale for each segment is taken from the main layer by default. But it can also be set separately for each level of the pie chart.

#### 6.3.8.2. Pie chart for single cluster

The symbology for displaying pie chart for the single cluster (the feature that did not appear in any cluster) is set via `eLiteGIS_clustering_singleFeatureDrawMode` parameter.

This parameter can have the following values:

- *Simple* – the symbol from the coloring of the source features without highlight is used;
- *Cluster* – the symbol is used as in the cluster with the caption "1" and highlighting from the pie chart;
- *Simplechart* – the symbol from the coloring of the source features is used with the highlight from the pie chart.

## 6.4. Subtypes

Subtypes – this is the data classification method where subgroups of features with the same attributes of a features class are used.

Subtypes allow you to:

- Set default values for the selected attribute, which will be automatically assigned to a new feature depending on the subtype to which it belongs;
- Group features of the same type by any attribute without the need to create separate feature classes, which improves the database performance;
- Apply reference books (domains) of coded values to features for each subtype;
- Create rules governing relationships between feature classes at the subtypes level.

eLiteGIS and CoGIS support subtypes set at the level of the map project in QGIS.

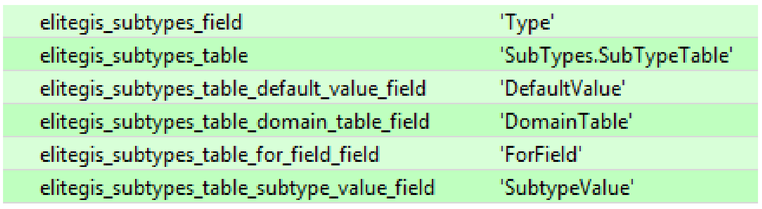
The settings for work with subtypes are specified via variables for the layer, see Table 5.

Table 5 – Variables for setting subtypes in the layer

| Variable | Description | By default |
|---|---|---|
| *eLiteGIS_subtypes_field* | *Name of field for subtypes* | - |
| *eLiteGIS_subtypes_table* | *Name of table in database with description of domains and default values (table of subtypes)* | - |
| *eLiteGIS_subtypes_table_default_value_field* | *Name of field in the table of subtypes with the default value for subtype* | *DefaultValue* |
| *eLiteGIS_subtypes_table_domain_table_field* | *Name of field in the table of subtypes with the name of the table containing domain values* | *DomainTable* |
| *eLiteGIS_subtypes_table_for_field_field* | *Name of field in the table of subtypes with the name of the field in the main table with features of the feature class where the domain values of subtypes should be shown* | *ForField* |
| *eLiteGIS_subtypes_table_subtype_value_field* | *Name of field in the table of subtypes where the value of the parent type is stored* | *SubtypeValue* |

Variables that have the default value are optional, provided that the fields in the subtype table are named appropriately.

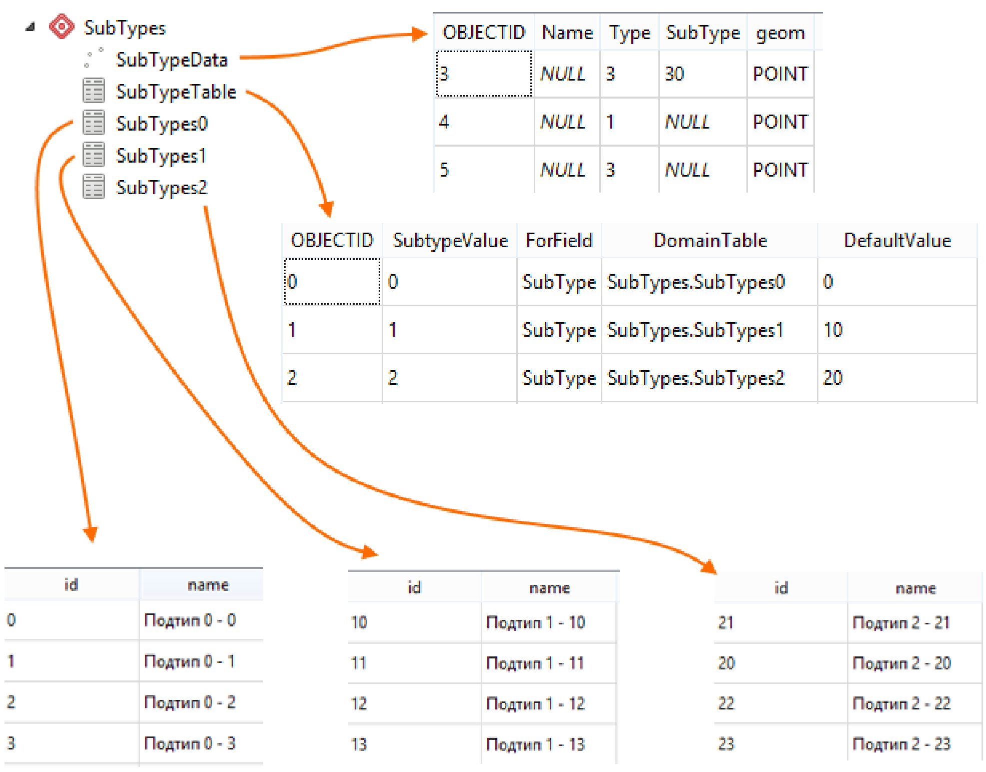
The example of filled variables for the table is shown in the Figure 100.



| | |
|---|---|
| elitegis_subtypes_field | 'Type' |
| elitegis_subtypes_table | 'SubTypes.SubTypeTable' |
| elitegis_subtypes_table_default_value_field | 'DefaultValue' |
| elitegis_subtypes_table_domain_table_field | 'DomainTable' |
| elitegis_subtypes_table_for_field_field | 'ForField' |
| elitegis_subtypes_table_subtype_value_field | 'SubtypeValue' |

Figure 100 – Example of parameters set for the subtypes in the layer

The example of table structure in the database and how they should be filled is shown on Figure 101.



| OBJECTID | Name | Type | SubType | geom |
|---|---|---|---|---|
| 3 | *NULL* | 3 | 30 | POINT |
| 4 | *NULL* | 1 | *NULL* | POINT |
| 5 | *NULL* | 3 | *NULL* | POINT |

| OBJECTID | SubtypeValue | ForField | DomainTable | DefaultValue |
|---|---|---|---|---|
| 0 | 0 | SubType | SubTypes.SubTypes0 | 0 |
| 1 | 1 | SubType | SubTypes.SubTypes1 | 10 |
| 2 | 2 | SubType | SubTypes.SubTypes2 | 20 |

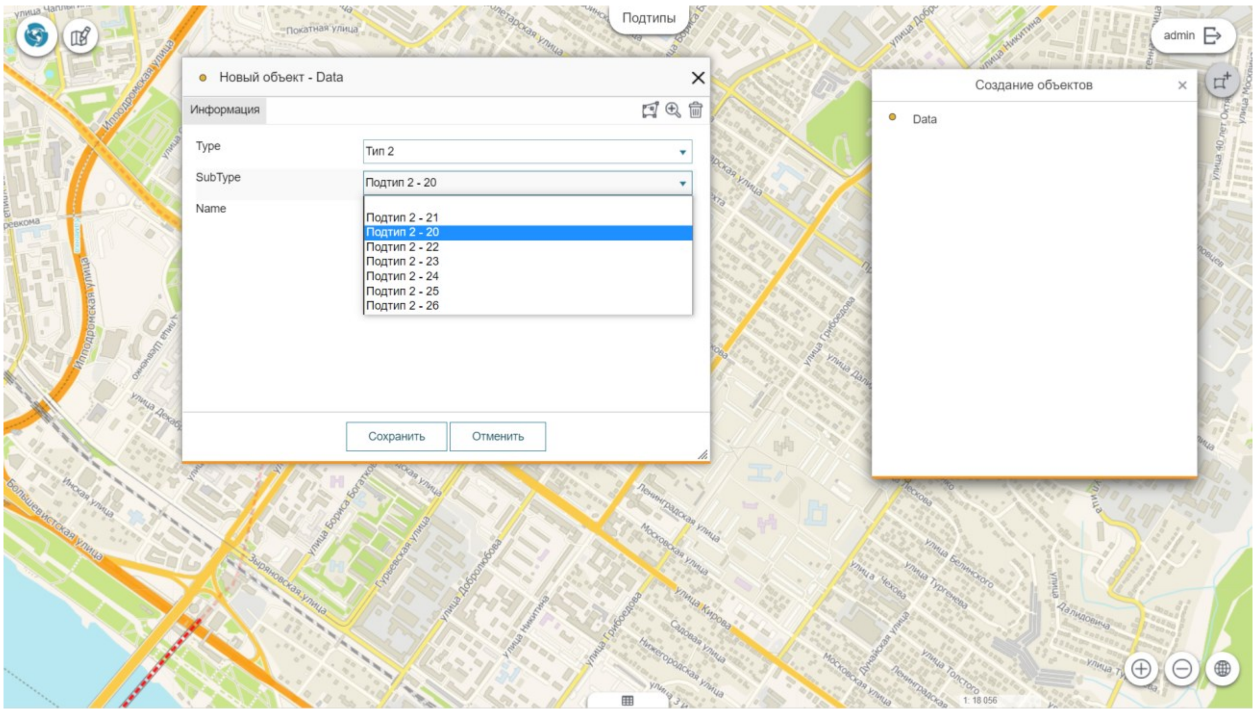
| id | name |
|---|---|
| 0 | Подтип 0 - 0 |
| 1 | Подтип 0 - 1 |
| 2 | Подтип 0 - 2 |
| 3 | Подтип 0 - 3 |

| id | name |
|---|---|
| 10 | Подтип 1 - 10 |
| 11 | Подтип 1 - 11 |
| 12 | Подтип 1 - 12 |
| 13 | Подтип 1 - 13 |

| id | name |
|---|---|
| 21 | Подтип 2 - 21 |
| 20 | Подтип 2 - 20 |
| 22 | Подтип 2 - 22 |
| 23 | Подтип 2 - 23 |

Figure 101 – Example of the structure and filling of tables in the database with subtypes

The example of the map with features, for one of the attributes of which subtypes are set, is shown on Figure 102. The figure demonstrates the dialog of the feature creation in which possible values for the *SybType* attribute are automatically substituted depending on the selected feature type (*Type*).



Новый объект - Data

Информация

| | |
|---|---|
| Type | Тип 2 |
| SubType | Подтип 2 - 20 |
| Name | |

Подтип 2 - 21
Подтип 2 - 20
Подтип 2 - 22
Подтип 2 - 23
Подтип 2 - 24
Подтип 2 - 25
Подтип 2 - 26

Figure 102 – Creating the new feature, taking into account the grouping of features by subtypes

## 6.5.Semi scale dependency

Semi scale dependency is the ability to set maximum and minimum scales and two sizes for a symbol, beyond which the symbol size does not change, but between which it changes linearly depending on the reduction/increase factors.

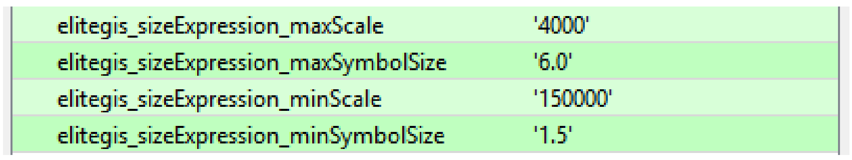
**eLiteGIS** supports settings of dependency of symbols on the scale, specified at the level of the map project in QGIS.

These settings are specified via variables for the layer, see Table 6.

Table 6 – Variables for setting the semi scale dependency of the layer symbols

| Variable | Description | Properties for |
|---|---|---|
| *eLiteGIS_sizeExpression_maxScale* | *Maximum scale value* | *Layer* |
| *eLiteGIS_sizeExpression_maxSymbolSize* | *Increase factor value* | *Layer* |
| *eLiteGIS_sizeExpression_minScale* | *Minimum scale value* | *Layer* |
| *eLiteGIS_sizeExpression_minSymbolSize* | *Reduction factor value* | *Layer* |

The example of filled values for variables is shown on Figure 103.



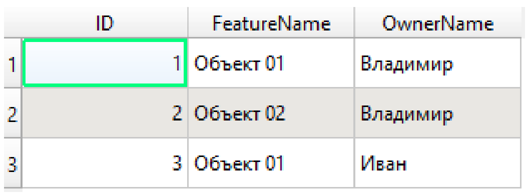
| | |
|---|---|
| elitegis_sizeExpression_maxScale | '4000' |
| elitegis_sizeExpression_maxSymbolSize | '6.0' |
| elitegis_sizeExpression_minScale | '150000' |
| elitegis_sizeExpression_minSymbolSize | '1.5' |

Figure 103 – Example of set variables for the layer display

## 6.6.Many-to-many relationship

Many-to-many relationship is the ability to set the relationship table for use of the many-to-many relationship at the level of the layers and/or the tables of the map project.

To enable support of this ability at the level of **eLiteGIS** and **CoGIS**, it is required to add the relationship table to the QGIS project, as shown on Figure 104.



| | ID | FeatureName | OwnerName |
|---|---|---|---|
| 1 | 1 | Объект 01 | Владимир |
| 2 | 2 | Объект 02 | Владимир |
| 3 | 3 | Объект 01 | Иван |

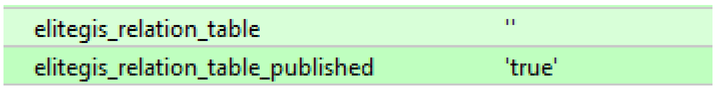
Figure 104 – Example of relationship table for setting the many-to-many relationship at the project's level

Next you need to set variables for this table, as shown in Table 7.

Table 7 – Variables for setting the many-to-many relationship table

| Variable | Description |
|---|---|
| *eLiteGIS_relation_table* | *Pointing to the table to use in relationship (empty value)* |
| *eLiteGIS_relation_table_published* | *Whether to publish the relationship table (true/false)* |

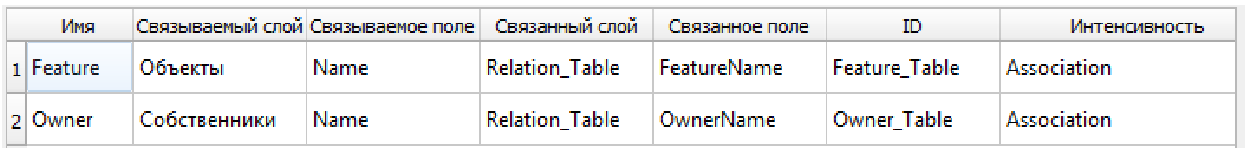
The example of filled values for the relationship table is shown on Figure 105.



| | |
|---|---|
| elitegis_relation_table | '' |
| elitegis_relation_table_published | 'true' |

Figure 105 – Example of filled values for the relationship table

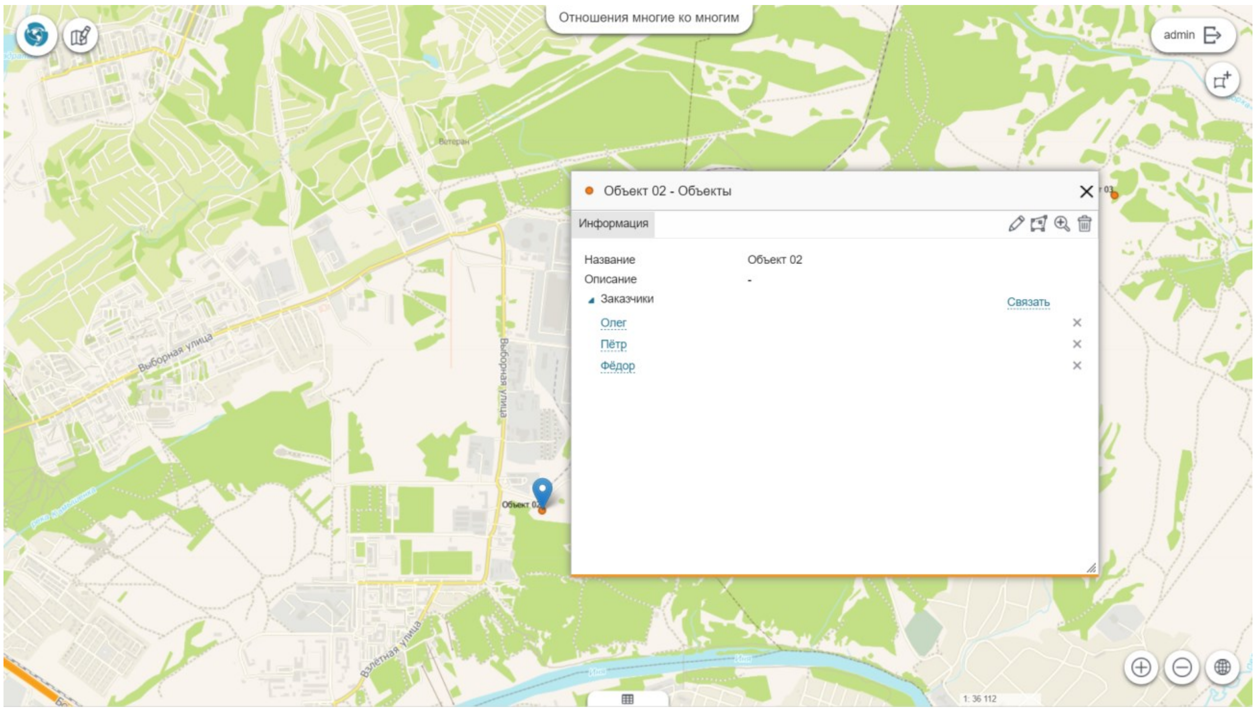
To finish settings, specify the relationship in the project properties for each related layer, see section 3.4 for details. The example of filled data about relationships in the project properties is shown on Figure 106.



| | Имя | Связываемый слой | Связываемое поле | Связанный слой | Связанное поле | ID | Интенсивность |
|---|---|---|---|---|---|---|---|
| 1 | Feature | Объекты | Name | Relation_Table | FeatureName | Feature_Table | Association |
| 2 | Owner | Собственники | Name | Relation_Table | OwnerName | Owner_Table | Association |

Figure 106 – Example of filled data about relationships in the project's properties

The figures below show the example of map containing the layer with features and the table with data about owners of these features, related by the many-to-many relationship, see Figure 107-Figure 109.



| | |
|---|---|
| Название | Объект 02 |
| Описание | - |

Заказчики — Связать
- Олег
- Пётр
- Фёдор

Figure 107 – Example of a feature related with many customers

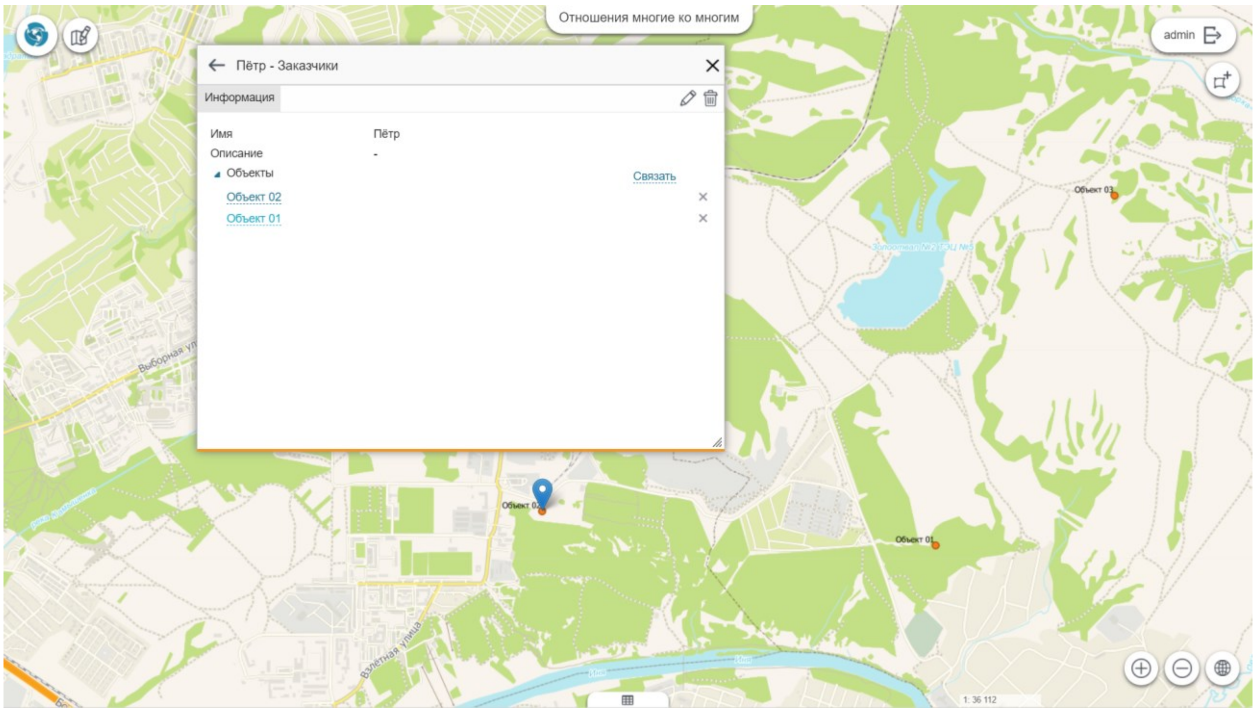


Figure 108 – Example of a customer related with many features

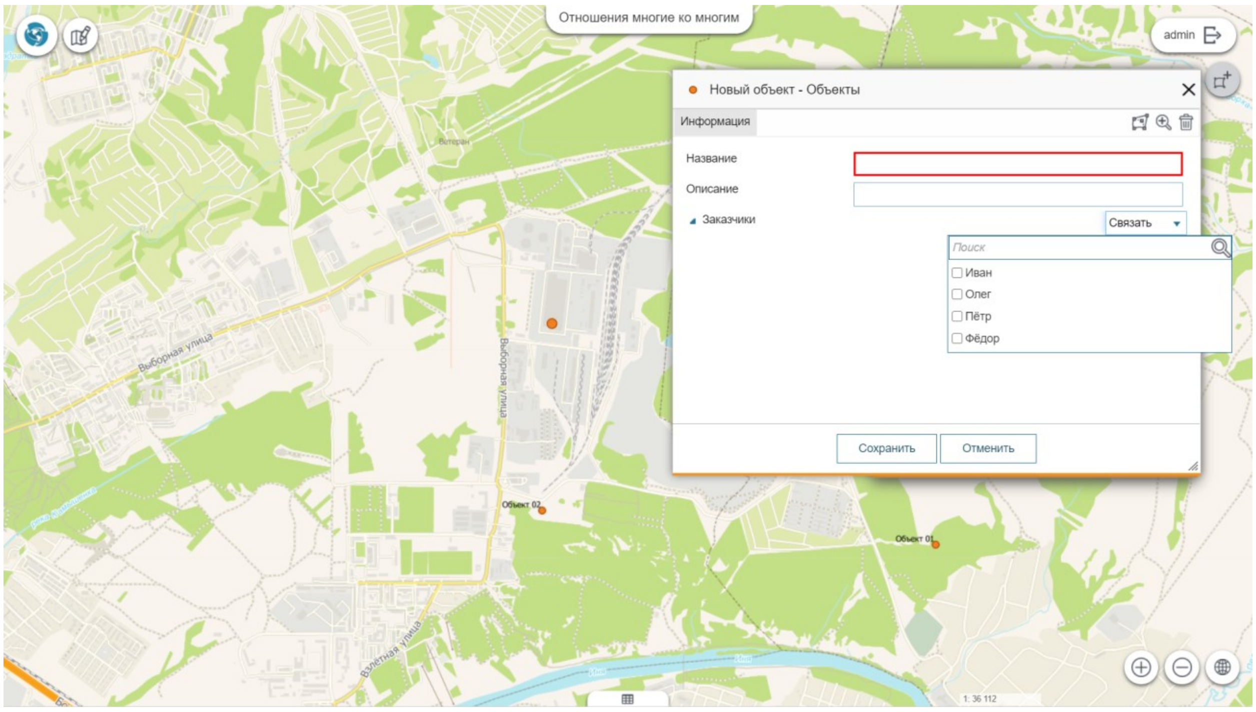


Figure 109 – Creation of the new feature with the option to relate it with one or multiple customers

As by adding the relationship table to the map project, for the variable `eLiteGIS_relation_table_published` the value *true* has been specified, then the relationship table can be also viewed in the published map application, see Figure 110.

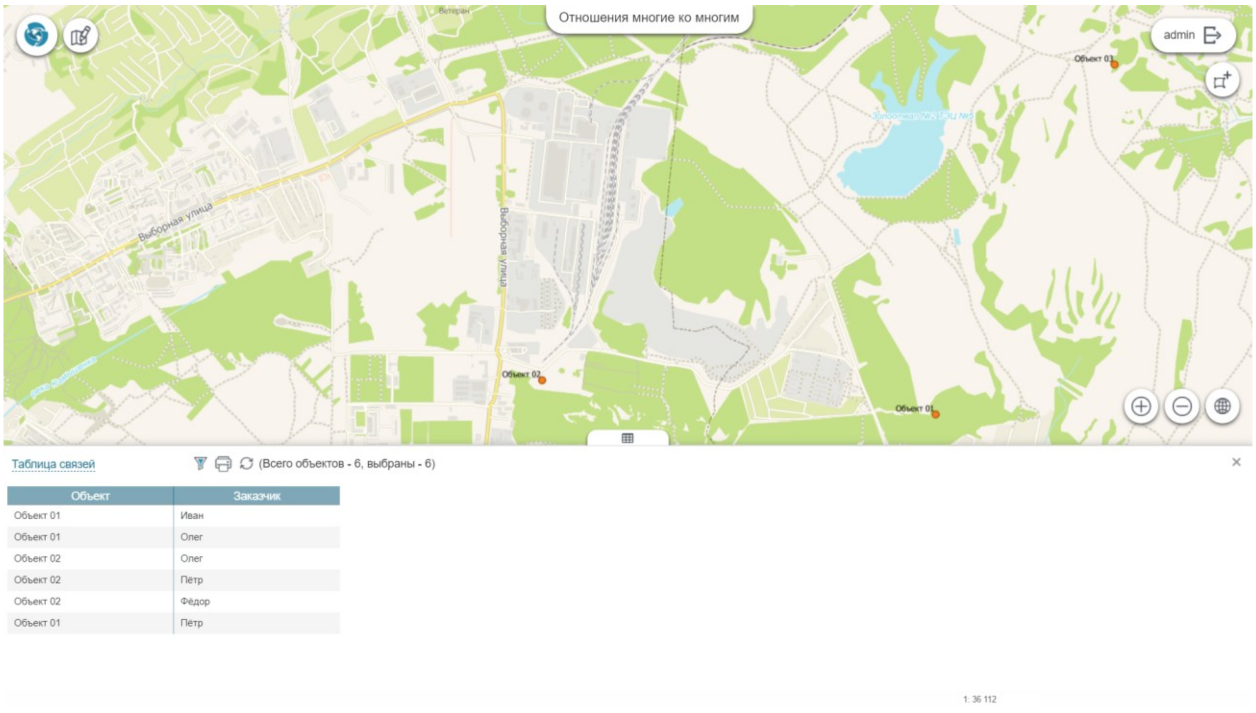


Таблица связей (Всего объектов - 6, выбраны - 6)

| Объект | Заказчик |
|---|---|
| Объект 01 | Иван |
| Объект 01 | Олег |
| Объект 02 | Олег |
| Объект 02 | Пётр |
| Объект 02 | Фёдор |
| Объект 01 | Пётр |

Figure 110 – Relationship table in the published map application

## 6.7.Annotations

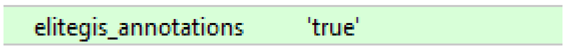
Enabling annotation support for a layer allows you to label polyline features as static labels. All properties for annotations are taken from the properties of the regular label, see section 5.3.

To enable annotation support at the level of **eLiteGIS**, you need to specify the appropriate value for the layer variable, see Table 8.

Table 8 – Variable for support for layer annotations

| Variable | Description |
|---|---|
| *eLiteGIS_annotations* | *Points that the feature should be labeled with the annotation (true/false)* |

The example of filled variable value for the layer is shown on Figure 112.



elitegis_annotations 'true'

Figure 111 – Example of filled variable value for annotation support

## 6.8.SQL query based view

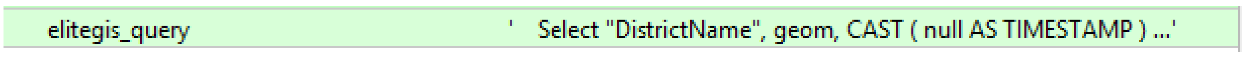
The SQL query based view is the ability to create layers in a project based on a SQL query to tables in a database. The property is set for the temporary layer.

To enable this option at the **eLiteGIS** level, it is necessary to specify the corresponding value for the variable for the temporary layer in the map project, see Table 9.

Table 9 – Variable for creation of the SQL query based view

| Variable | Description |
|---|---|
| *eLiteGIS_query* | *SQL query to table in database* |

The example of filled variable value for the temporary layer is shown on Figure 112.



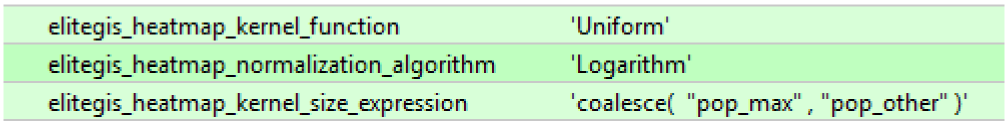
| | |
|---|---|
| elitegis_query | ' Select "DistrictName", geom, CAST ( null AS TIMESTAMP ) ...' |

Figure 112 – Example of filled value for creation of the SQL query based view

## 6.9.Interpolation maps

Interpolation maps are provided to visualize clusters of point data and/or identify high concentrations of activity.

At the level of the selected layer in QGIS project it is possible to set variables that will allow you to display the layer data as the interpolation map by publishing the project in **eLiteGIS**, see Figure 113. At that, no additional service setting at the level of **eLiteGIS** is required.



| | |
|---|---|
| elitegis_heatmap_kernel_function | 'Uniform' |
| elitegis_heatmap_normalization_algorithm | 'Logarithm' |
| elitegis_heatmap_kernel_size_expression | 'coalesce( "pop_max" , "pop_other" )' |

Figure 113 – Variables for displaying the layer data as interpolation map

The variable `eLiteGIS_heatmap_normalization_algorithm` is provided for setting of normalization of the values calculation and can have the following values, see Table 10.

Table 10 – Possible values for variable eLiteGIS_heatmap_normalization_algorithm

| Value | Description |
|---|---|
| *Linear (by default)* | *The calculated value is normalized linearly (uniformly)* |
| *Logarithm* | *The calculated value is normalized by the logarithm, the average values are close to the maximum* |

The variable `eLiteGIS_heatmap_kernel_function` is provided for setting rendering parameters (kernel function) and can have the following values, see **Ошибка! Источник ссылки не найден.**.

Table 11 – Possible values for variable eLiteGIS_heatmap_kernel_function

| Value | Description |
|---|---|
| *Uniform (by default)* | *Uniform distribution* |
| *Triangular* | *Triangular distribution* |
| *Epanechnikov* | *Epanechnikov (parabolic) distribution* |
| *Quartic* | *Quartic distribution* |
| *Triweight* | *Triweight distribution* |
| *Tricube* | *Tricube distribution* |
| *Cosine* | *Cosine distribution* |

The graphs of the mentioned kernel functions are shown on Figure 114.

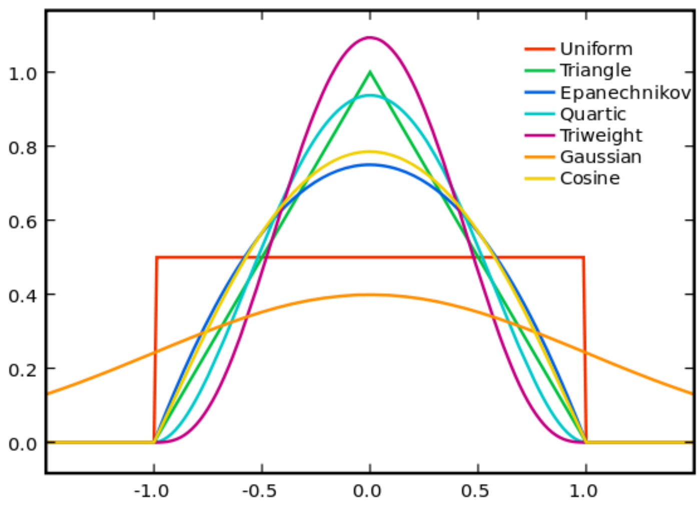


Figure 114 – Graphs of kernel functions for use by rendering of interpolation map

The variable `eLiteGIS_heatmap_kernel_size_expression` is provided for setting parameter of kernel radius calculation by SQL expression (instead of standard parameter Radius in QGIS). The variable takes the value as SQL expression for calculation of the radius value.

The variable `eLiteGIS_heatmap_model` is provided for setting parameters for rendering of interpolation map via the interpolation algorithm. The possible values of this variable are shown in **Ошибка! Источник ссылки не найден.**.

Table 12 – Possible values for variable eLiteGIS_heatmap_model

| Value | Description |
|---|---|
| *Accumulation* | *Interpolation mode is off, it is equivalent to the absence of this property* |
| *Interpolation* | *Interpolation mode is on* |

If interpolation is used, the variables for kernel function and kernel size (`eLiteGIS_heatmap_normalization_algorithm` and `eLiteGIS_heatmap_kernel_size_expression`) are not considered.

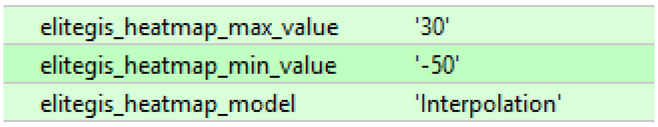
Additionally, the interpolation map can be set by specifying value for normalization window, see Table 13.

Table 13 – Variables for normalization window

| Variable | Description |
|---|---|
| *eLiteGIS_heatmap_min_value* | *Minimum value of normalization window* |
| *eLiteGIS_heatmap_max_value* | *Maximum value of normalization window* |

These parameters are optional. If they are not specified, the minimum value is taken as null, and the maximum value is taken from the *Maximum value* parameter in QGIS.

The example of filled variables values for the normalization window and the interpolation algorithm is shown on Figure 115.



| | |
|---|---|
| elitegis_heatmap_max_value | '30' |
| elitegis_heatmap_min_value | '-50' |
| elitegis_heatmap_model | 'Interpolation' |

Figure 115 – Example of filled variables values for the normalization window and the interpolation algorithm

The example of interpolation map published using **eLiteGIS** and **CoGIS** based on the map project described above, is shown on Figure 116.

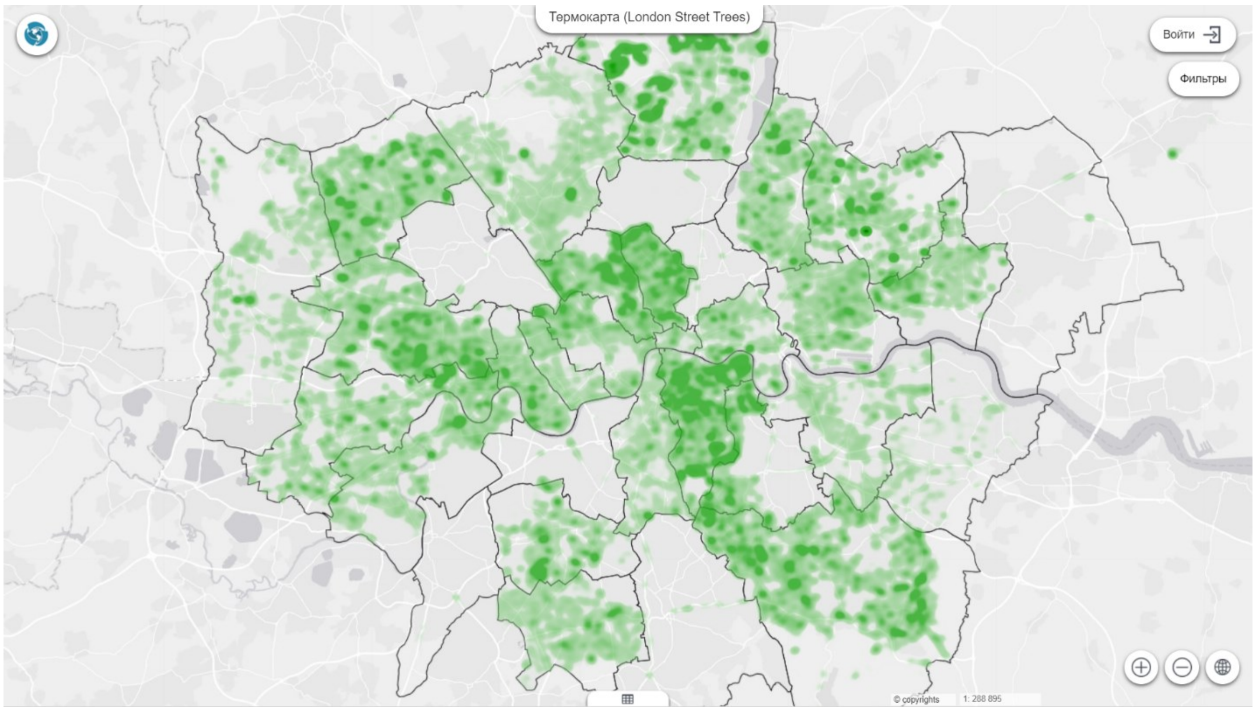


Figure 116 – Interpolation map showing distribution of trees in London districts

## 6.10. Heat maps

Heat maps are "temperature" maps of the territory, showing a continuous distribution of the values of a feature (air temperature, atmospheric pressure, altitude, etc.), for the construction of which various interpolation algorithms are used.

Figure 117 and Figure 118 show heat maps on air temperature and precipitation in the territory of Russian Federation.

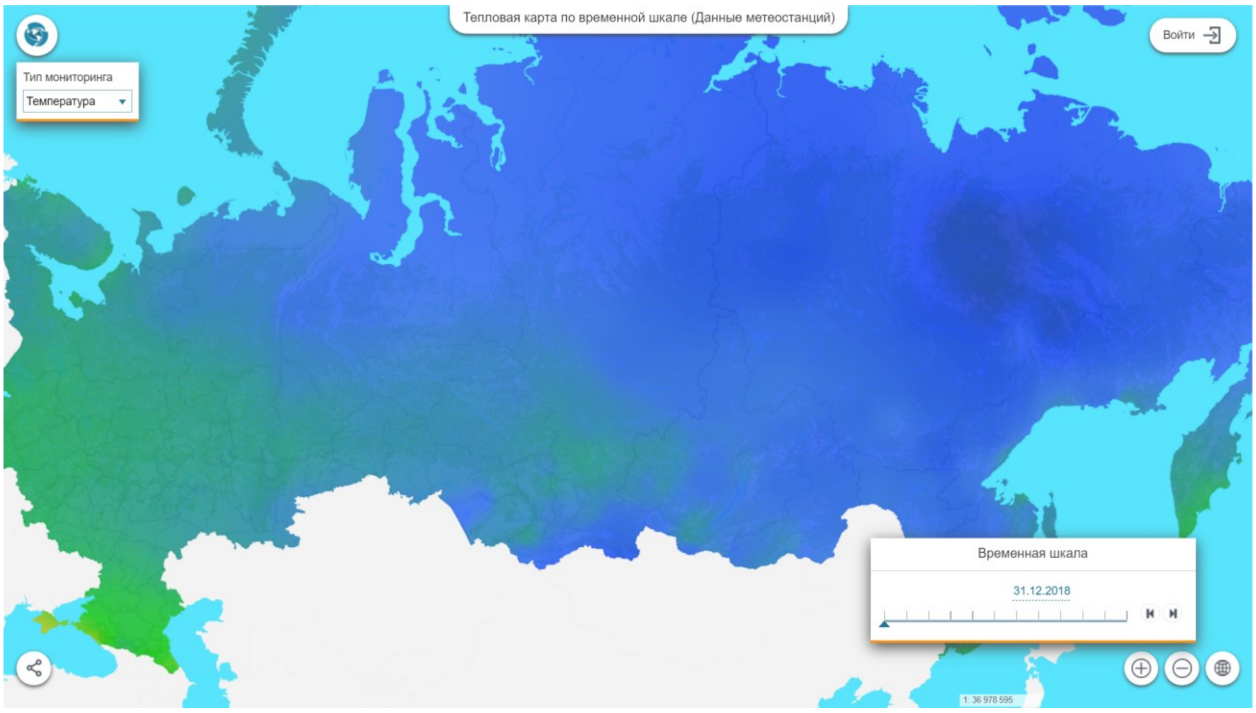


Figure 117 – Heat map on air temperature for specific period in the territory of Russian Federation

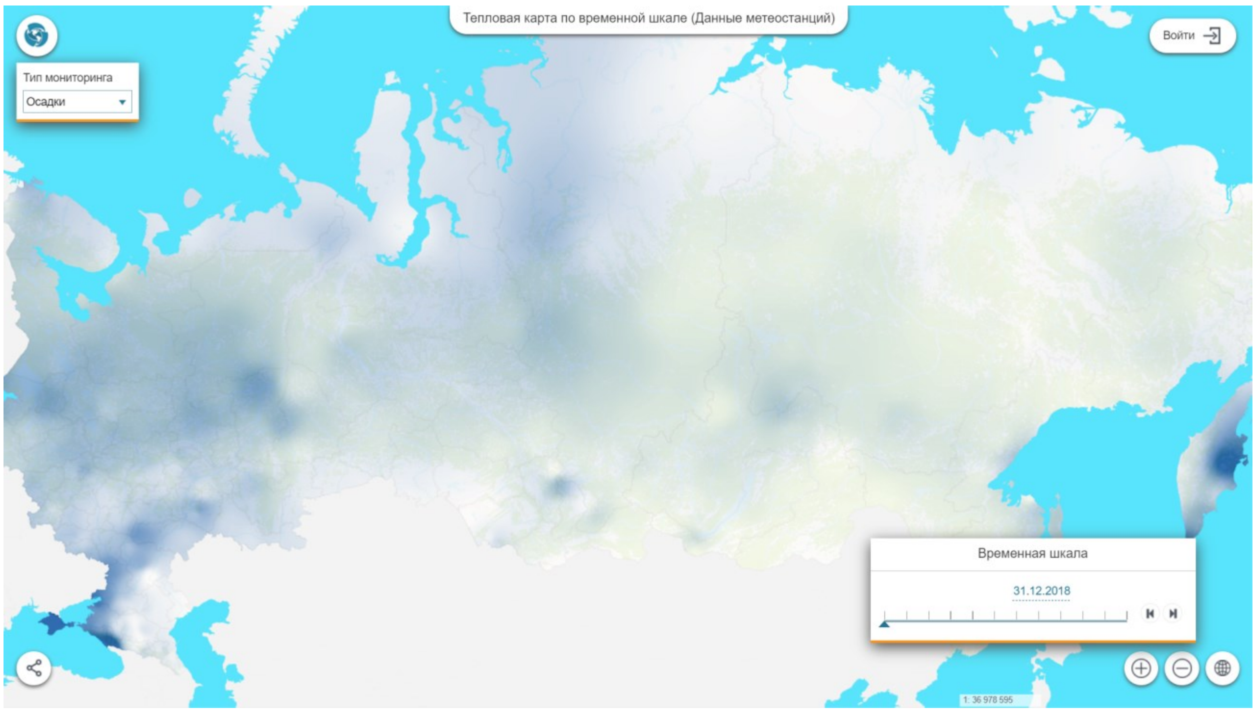


Figure 118 – Heat map on precipitation for specific period in the territory of Russian Federation

The heat maps were created based on weather station readings (see Figure 119) and interpolation of obtained values to the entire territory.

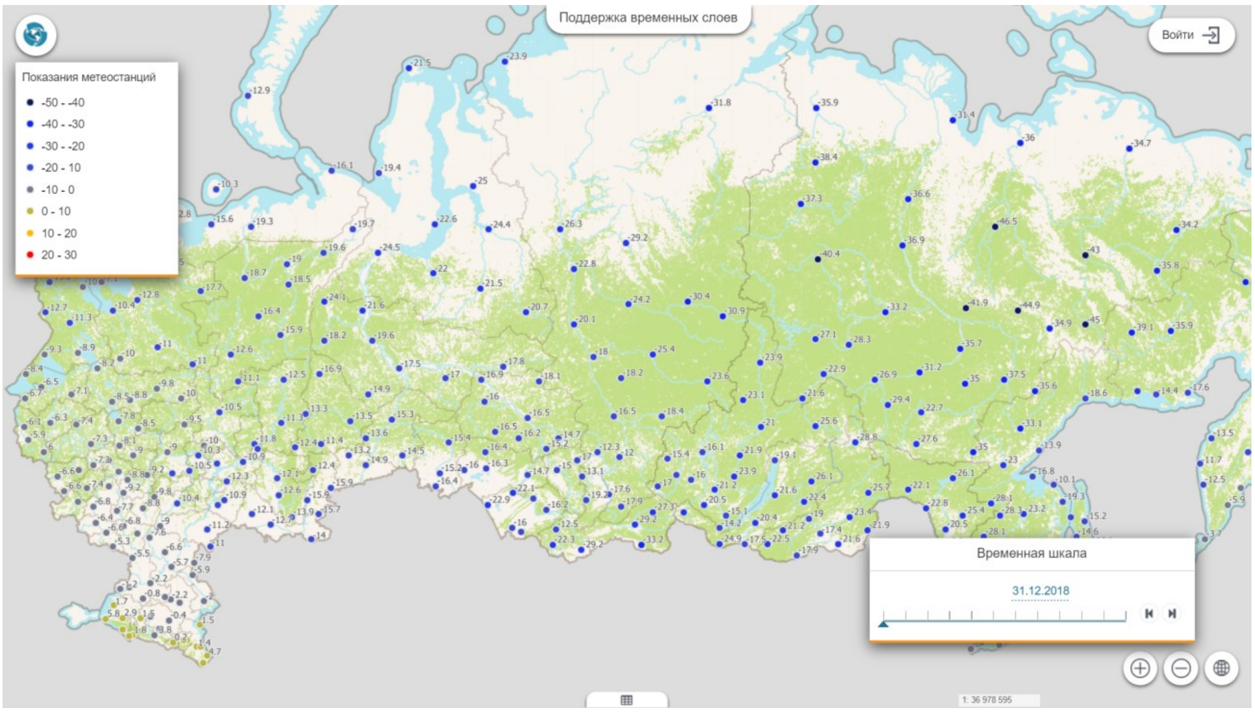


Figure 119 – Source data from weather station based on which the heat maps were created

At the level of the selected layer in QGS project it is possible to set variables that will allow you to display the layer data as heat map by publishing the project in **eLiteGIS**:

- `eLiteGIS_heatmap_model` – enables interpolation mode for the layer;
- `eLiteGIS_heatmap_min_value` – minimum value of normalization window;
- `eLiteGIS_heatmap_max_value` – maximum value of normalization window.

At that, no additional setting of service at **eLiteGIS** level is required.

The variable `eLiteGIS_heatmap_model` is provided for setting parameters for rendering of heat map via the interpolation algorithm. The possible values of this variable are shown in Table 14.

Table 14 – Possible values for variable eLiteGIS_heatmap_model

| Value | Description |
|---|---|
| *Accumulation* | *Interpolation mode is off, it is equivalent to the absence of this property* |
| *Interpolation* | *Interpolation mode is on* |

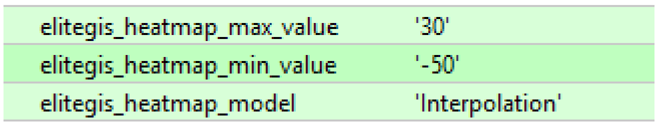
Additionally, the heat map can be set by specifying value for normalization window, see Table 15.

Table 15 – Variables for normalization window

| Переменная | Description |
|---|---|
| *eLiteGIS_heatmap_min_value* | *Minimum value of normalization window* |
| *eLiteGIS_heatmap_max_value* | *Maximum value of normalization window* |

These parameters are optional. If they are not specified, the minimum value is taken as null, and the maximum value is taken from the *Maximum value* parameter in QGIS.

The example of filled variables values for the normalization window and the interpolation algorithm is shown on Figure 120.



|  |  |
|---|---|
| elitegis_heatmap_max_value | '30' |
| elitegis_heatmap_min_value | '-50' |
| elitegis_heatmap_model | 'Interpolation' |

Figure 120 – Example of filled variables values for the normalization window and the interpolation algorithm

Examples of heat maps published using **eLiteGIS** and **CoGIS** based on the above mentioned map project are shown on Figure 117 and Figure 118.

## 6.11. Pseudo 3D

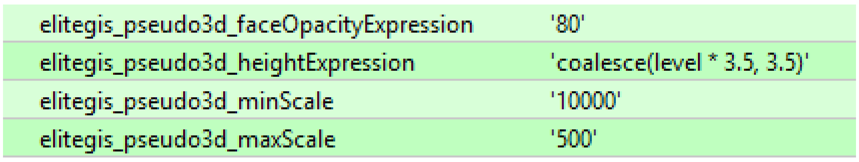
Pseudo 3D is used to display pseudo 3D shapes based on areal features with a given height in meters. The most common use of this feature is to display the height of buildings based on their number of stores.

At the level of the selected layer in the QGS project, you can set variables that will allow you to display layer objects as pseudo 3D features when publishing a project in **eLiteGIS**, see Figure 113. At that, no additional setting of service at **eLiteGIS** level is required.

Table 16 – Variables for display of features as pseudo 3D shapes

| Variable | Description |
|---|---|
| *eLiteGIS_pseudo3d_heightExpression* | Feature's height in meters, *sql expression is allowed* |
| *eLiteGIS_pseudo3d_faceOpacityExpression* | *Opacity of side faces in percent (0-100), calculated from the feature's opacity value specified in the symbol coloring* |
| *eLiteGIS_pseudo3d_minScale* | *Minimum visibility scale of pseudo 3D for the feature (optional)* |
| *eLiteGIS_pseudo3d_maxScale* | *Maximum visibility scale of pseudo 3D for the feature (optional)* |

The example of filled values for variables is shown on Figure 121.



|  |  |
|---|---|
| elitegis_pseudo3d_faceOpacityExpression | '80' |
| elitegis_pseudo3d_heightExpression | 'coalesce(level * 3.5, 3.5)' |
| elitegis_pseudo3d_minScale | '10000' |
| elitegis_pseudo3d_maxScale | '500' |

Figure 121 – Example of filled values for variables

The example of pseudo 3D visualization based on a service published with **eLiteGIS** and **CoGIS** and configured at the map project level as described above is shown on Figure 122.

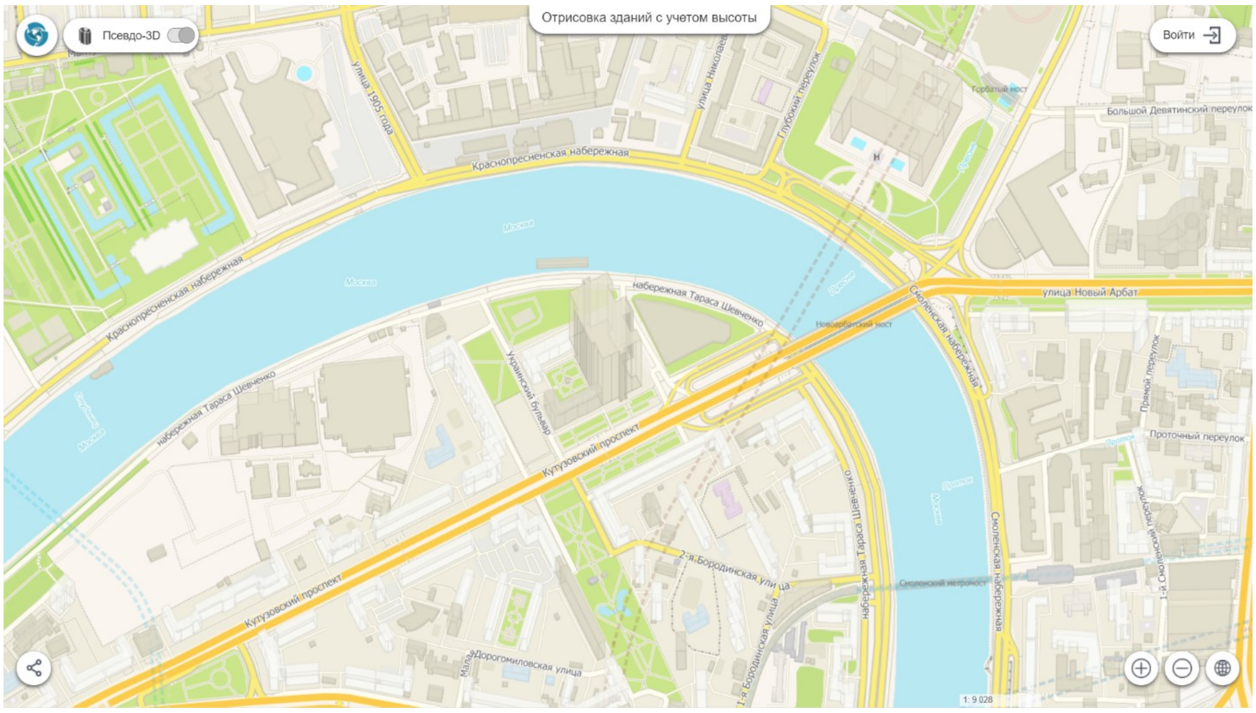


Figure 122 – Example of pseudo 3D visualization

## 6.12. Number of layers

In the layer properties in QGIS it is possible to set the identifier (id) for the selected layer. This ID will be assigned to the layer when publishing in **eLiteGIS** instead of the automatically assigned ID.

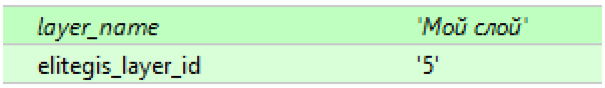
This property can be useful when debugging of QGS project, based on which the map service is published and maps are created in **CoGIS**, as the order, quantity and IDs of layers, respectively, can change, but at that used in the settings of map in **CoGIS**.

The layer identifier is set via variable `eLiteGIS_layer_id`.

At that the order of layers remains the same as in QGIS project.

Setting two similar numbers is not allowed.

The example of filled value for variable is shown on Figure 123.



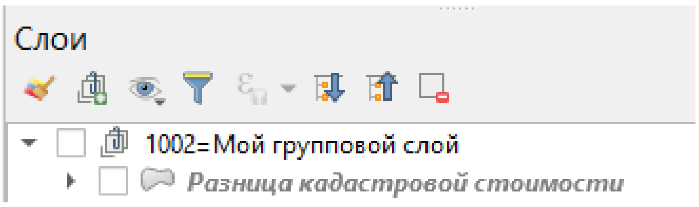
| | |
|---|---|
| *layer_name* | *'Мой слой'* |
| elitegis_layer_id | '5' |

Figure 123 – Example of filled value for variable which specifies the ID of the selected layer

The number for Group layer is set in the layer name as follows: *number=layer name*

For example: *1002=My group layer.*

The example of group layer with specified number in QGS project is shown on Figure 124.



1002=Мой групповой слой

*Разница кадастровой стоимости*

Figure 124 – Example of group layer with specified number

## 6.13. Cropping map using mask layer

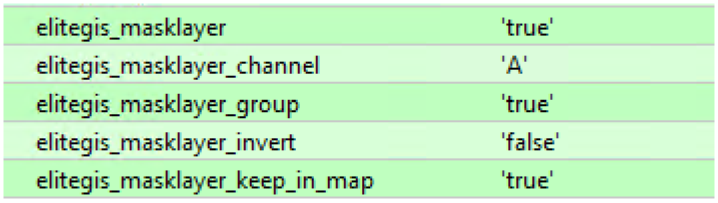
At the level of the QGS project you can configure cropping of all layers of the map service with a special layer called a mask (MaskLayer). The mask layer can be of any type (point, polyline, polygon) and of any color.

The mask layer can be any layer of the map project. The following layer variables are used for setting the mask layer, see Table 16.

Table 17 – Variables for setting mask layer

| Variable | Description | Values |
|---|---|---|
| *eLiteGIS_masklayer* | *Indicates whether the layer is the mask layer* | *true/false* |
| *eLiteGIS_masklayer_channel* | *Indicates which channel should be used as the mask (in most cases, the A opacity is used)* | ***A (opacity)***<br>*R (red)*<br>*G (green)*<br>*B (blue)* |
| *eLiteGIS_masklayer_group* | *Cropping by group layer. If the current layer is in the group layer, then all layers from this group will be used with this cropping property.* | *true/**false*** |
| *eLiteGIS_masklayer_invert* | *Indicates whether the mask inversion should be used* | *true/**false*** |
| *eLiteGIS_masklayer_keep_in_map* | *Indicates whether to display MaskLayer in the map service* | *true/false* |

The example of filled values for variables is shown on Figure 125.



| | |
|---|---|
| elitegis_masklayer | 'true' |
| elitegis_masklayer_channel | 'A' |
| elitegis_masklayer_group | 'true' |
| elitegis_masklayer_invert | 'false' |
| elitegis_masklayer_keep_in_map | 'true' |

Figure 125 – Example of filled values for setting the mask layer

## 6.14. Power line symbol

In topographic maps, to display power lines, the corresponding symbol is used: a linear feature, at each vertex of which a symbol of a power line pole (point/square/rectangle) and indented arrows in the directions of the line are displayed, see Figure 126.

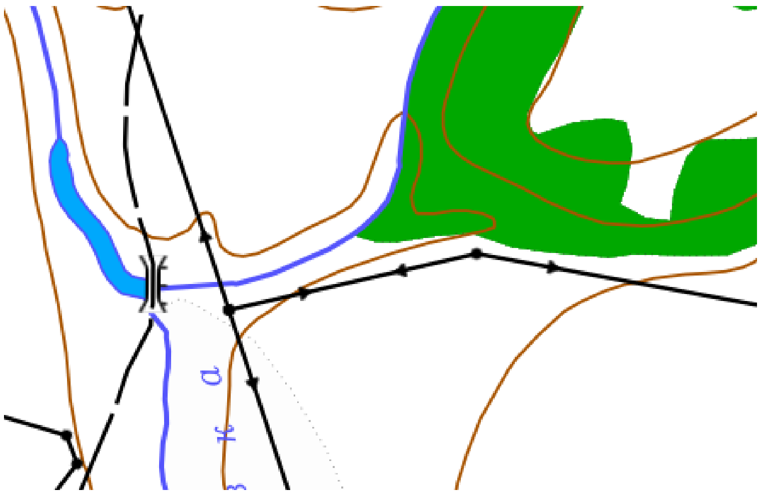


Figure 126 – Example of displaying power line on the topographic map

For setting such symbol in QGIS you need to create a combined linear symbol as shown on Figure 127.

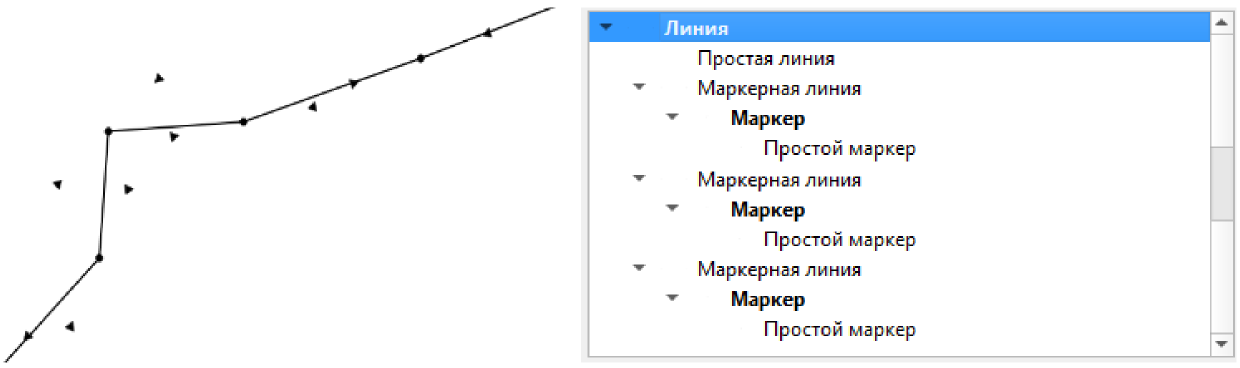
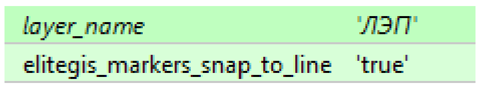

Линия
Простая линия
Маркерная линия
Маркер
Простой маркер
Маркерная линия
Маркер
Простой маркер
Маркерная линия
Маркер
Простой маркер

Figure 127 – Setting combined symbol for power line

In the layer parameters specify the variable `eLiteGIS_markers_snap_to_line` as *true*, see Figure 128.



| | |
|---|---|
| *layer_name* | *'ЛЭП'* |
| elitegis_markers_snap_to_line | 'true' |

Figure 128 – Example of filled variable value, that allows you to display complex combined symbols (such as power line) on the map

Scale dependency is possible only if the marker itself is scale dependent. But at the same time, its size will also change.

# 7. Creating geocoding service

GIS server **eLiteGIS** allows you to create geocoding service by map service.

In general case, the geocoding service in **eLiteGIS** can be used not only on address data for matching addresses and coordinates, but also on any other data as the universal service for the free text search.

Before publishing the geocoding service, you need to make sure that the source data for this service is properly configured:

- the layers and fields in the layers, which will be searched, are defined in the database
- for layers where several fields will be used in the search, the values of these fields have been merged
- the index for the selected fields is built
- the map project is created from the selected layers
- in the layers properties in the map project, the display field is specified, which matches the fields by which the search will be carried out.

To build the address geocoder, it is necessary that in the map project, based on which the geocoding service will be created, the layers of buildings and streets are present.

This section provides detailed instructions on publishing geocoding service.

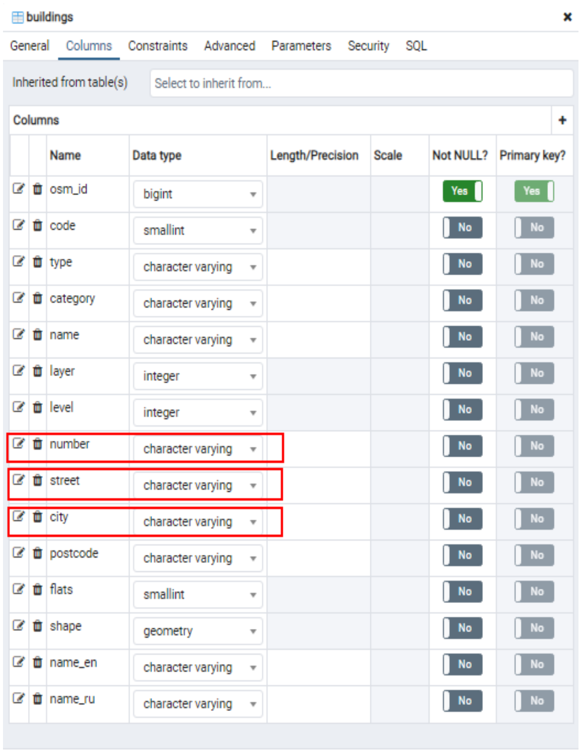
## 7.1. Preparing data in database

To create the geocoding service, you need to define the list of tables and fields in the PostgreSQL database that will be used for geocoding.

*Note: for work with tables and fields, the pgAdmin is used.*

Let's assume that for work of our service the layers of building and roads are created.

In the buildings layer the fields *city*, *street* and *number* will be used (see Figure 129). In the roads layer the field *name* will be used (see Figure 130).



buildings

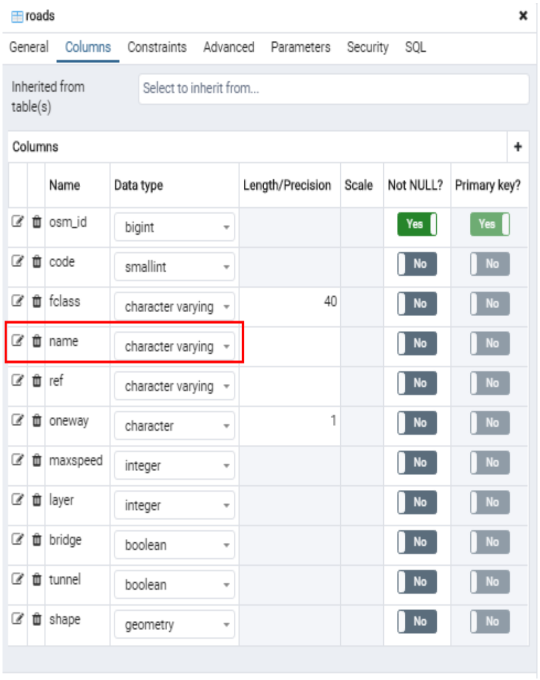
General | Columns | Constraints | Advanced | Parameters | Security | SQL

Inherited from table(s): Select to inherit from...

Columns

| Name | Data type | Length/Precision | Scale | Not NULL? | Primary key? |
|---|---|---|---|---|---|
| osm_id | bigint | | | Yes | Yes |
| code | smallint | | | No | No |
| type | character varying | | | No | No |
| category | character varying | | | No | No |
| name | character varying | | | No | No |
| layer | integer | | | No | No |
| level | integer | | | No | No |
| number | character varying | | | No | No |
| street | character varying | | | No | No |
| city | character varying | | | No | No |
| postcode | character varying | | | No | No |
| flats | smallint | | | No | No |
| shape | geometry | | | No | No |
| name_en | character varying | | | No | No |
| name_ru | character varying | | | No | No |

Figure 129 – Buildings layer: fields for geocoding



roads

General | Columns | Constraints | Advanced | Parameters | Security | SQL

Inherited from table(s): Select to inherit from...

Columns

| Name | Data type | Length/Precision | Scale | Not NULL? | Primary key? |
|---|---|---|---|---|---|
| osm_id | bigint | | | Yes | Yes |
| code | smallint | | | No | No |
| fclass | character varying | 40 | | No | No |
| name | character varying | | | No | No |
| ref | character varying | | | No | No |
| oneway | character | 1 | | No | No |
| maxspeed | integer | | | No | No |
| layer | integer | | | No | No |
| bridge | boolean | | | No | No |
| tunnel | boolean | | | No | No |
| shape | geometry | | | No | No |

Figure 130 – Roads layer: fields for geocoding

For correct search by the geocoder using the freetext index, for layers in which more than one field will be used, it is necessary to combine the values of these fields.

For example, for the buildings layer, you need to create a new search field and add the necessary values there as follows:

```
ALTER TABLE public.buildings ADD COLUMN search_text character varying;
UPDATE public.buildings SET search_text = TRANSLATE(COALESCE ("city", '') || ' ' ||
COALESCE ("street", '') || ' ' || COALESCE ("number", ''), './+-', '    ');
```

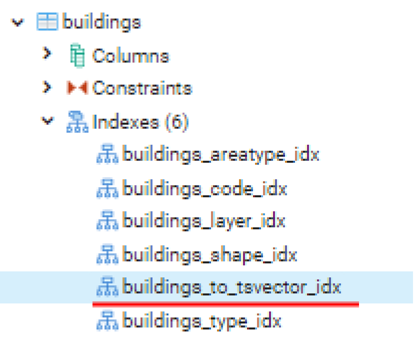
Then create the trigger to update this field, if needed:

```
CREATE FUNCTION buildings.update_search_text() RETURNS trigger AS $$
  BEGIN
    NEW.search_text = translate(NEW.city || ' ' || NEW.street || ' ' || NEW.number, './+-', '    ');
    RETURN NEW;
  END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER buildings_update_search_text BEFORE INSERT OR UPDATE ON
public.buildings
FOR EACH ROW EXECUTE PROCEDURE buildings.update_search_text();
```

And then specify the freetext index for the selected fields:

```
CREATE INDEX ON public.buildings
USING gin(to_tsvector('russian', COALESCE("search_text", '')));

CREATE INDEX ON public.roads
USING gin(to_tsvector('russian', COALESCE("name", '')));
```

After this it is recommended to make sure that the indices are created. To do so, check the information about the table in the database, as shown on Figure 131.



```
buildings
  Columns
  Constraints
  Indexes (6)
    buildings_areatype_idx
    buildings_code_idx
    buildings_layer_idx
    buildings_shape_idx
    buildings_to_tsvector_idx
    buildings_type_idx
```

Figure 131 – Buildings layer: indices for geocoding

## 7.2. Preparing the project in QGIS

For publishing of geocoding service in **eLiteGIS** it is needed to create map project with the geocoder data.

To do so, add the buildings and roads layers created on the previous step, to the map project, see Figure 132.

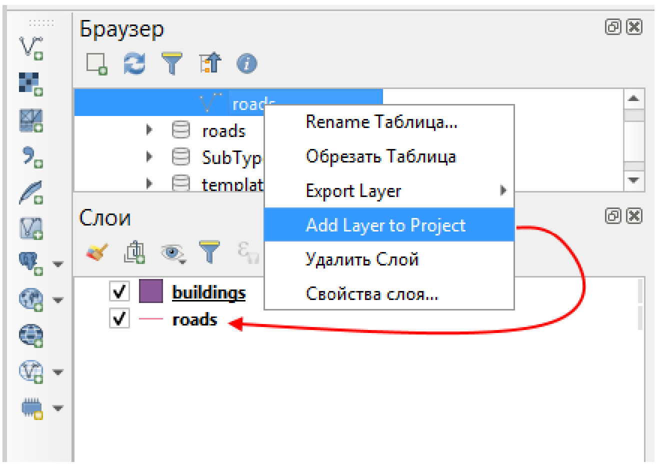


Figure 132 – Adding layers for geocoder to the QGIS map project

Next, for each layer, you need to specify the presence and correct order of fields. To do this, from the context menu of the layer, you need to open the attribute table, as shown on Figure 133.

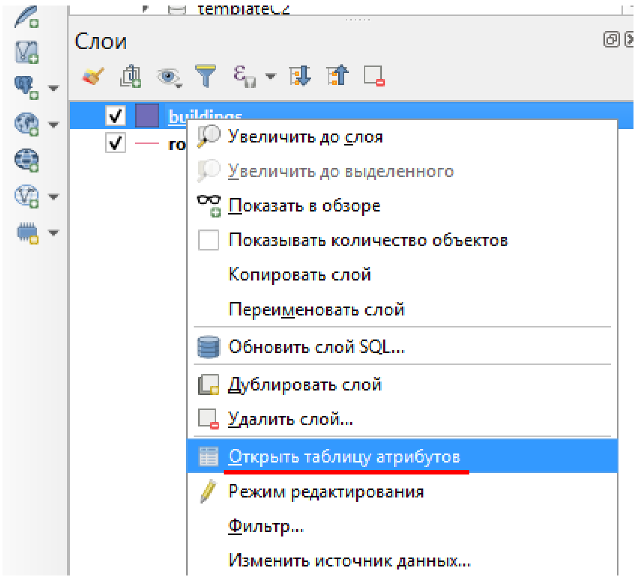
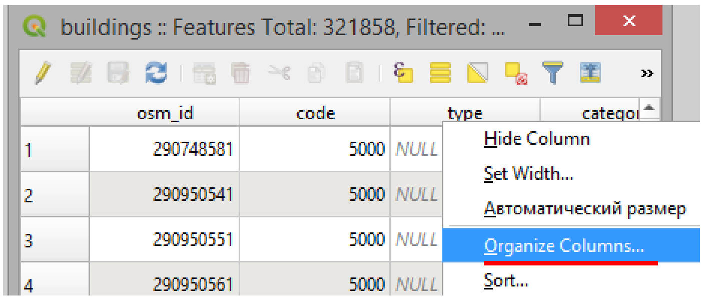


Figure 133 – Attribute table of the layer

Next, in the attribute table, from the context menu for the column headings, select *Organize Columns*, see Figure 134.



| | osm_id | code | type | categoı |
|---|---|---|---|---|
| 1 | 290748581 | 5000 | NULL | |
| 2 | 290950541 | 5000 | NULL | |
| 3 | 290950551 | 5000 | NULL | |
| 4 | 290950561 | 5000 | NULL | |

Figure 134 – Setting the fields composition

And then in the appeared window *Organize Table columns*, you need to leave enabled the field *primary_key* and those fields that will be used in the geocoding, see Figure 135. Press *OK.*

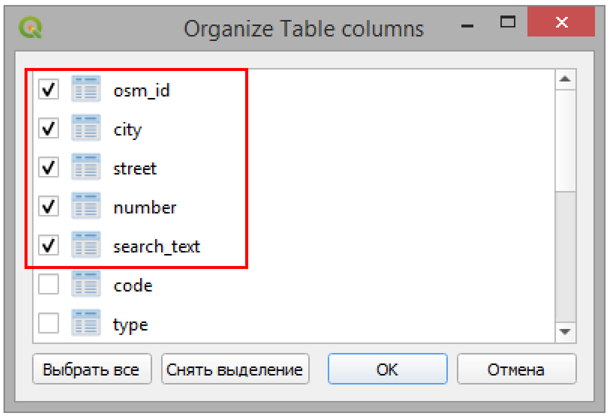
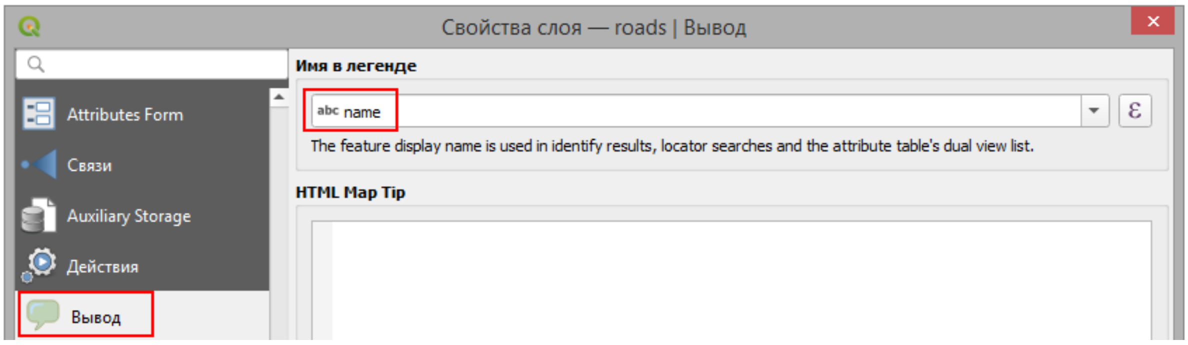


Figure 135 – Selection of layer fields for use in the geocoding service

This way the order of the fields in the table will be set. These steps must be repeated for all other layers.

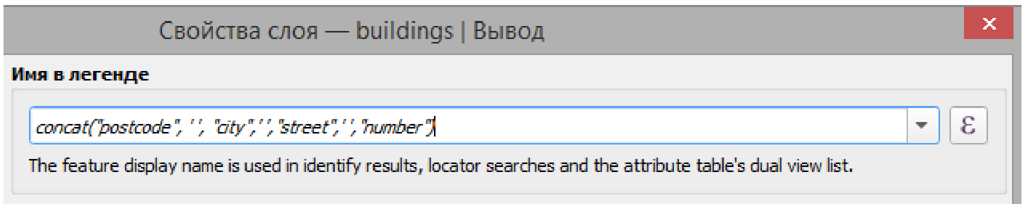
Then you need to give each layer a field that will be used as *Display Field*. To do this, select the *Display* item in the layer properties and set the corresponding field, see Figure 136 with the example.



The feature display name is used in identify results, locator searches and the attribute table's dual view list.

Figure 136 – Setting display field

If you need to set display by multiple fields, set *Display expression*. For example, for the buildings layer, the expression may look as following: *concat("postcode", ' ', "city",' ',"street",' ',"number")*, see Figure 137.



concat("postcode", ' ', "city",' ',"street",' ',"number")

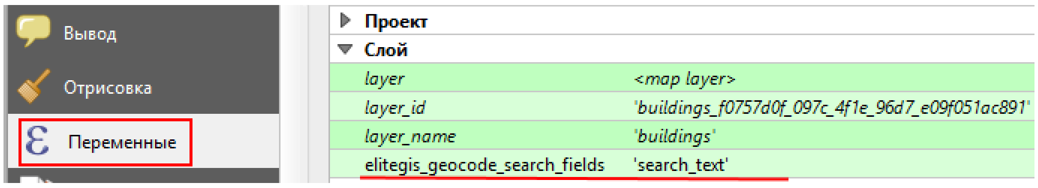
The feature display name is used in identify results, locator searches and the attribute table's dual view list.

Figure 137 – Display by multiple fields

*Note: only following functions are supported: "sin", "cos", "tan", "atan", "abs", "asin", "acos", "log", "log10", "cailing", "floor", "round", "ltrim", "rtrim", "substr", "substring", "concat", "lower", "upper", "pow", "andbits", "len", "length", "coalesce", "mod", "scale_linear", "scale_exp", "tostring"*

By default the freetext search is done by the field specified in *Display Field*.

If the expression is specified, the freetext search will not work. In this case, it is necessary to specify the field, by which the search should be done, by a separate property in the layer settings. To do this, in the layer properties in the Variables tab specify variable `eLiteGIS_geocode_search_fields` with the search field name, see Figure 138.



| | |
|---|---|
| layer | <map layer> |
| layer_id | 'buildings_f0757d0f_097c_4f1e_96d7_e09f051ac891' |
| layer_name | 'buildings' |
| elitegis_geocode_search_fields | 'search_text' |

Figure 138 – Setting search field

Now save the created QGS project, see Figure 139.

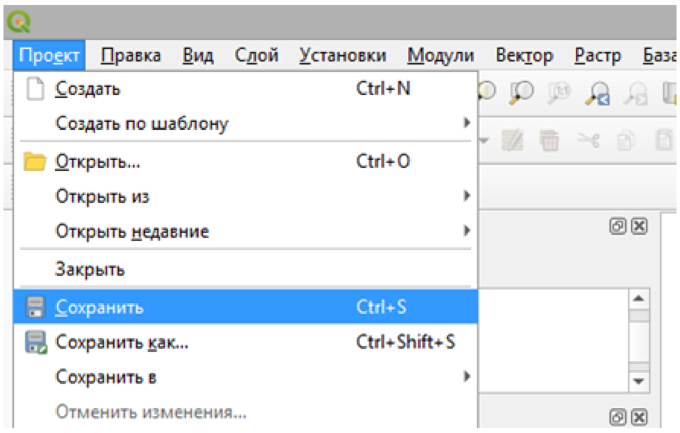


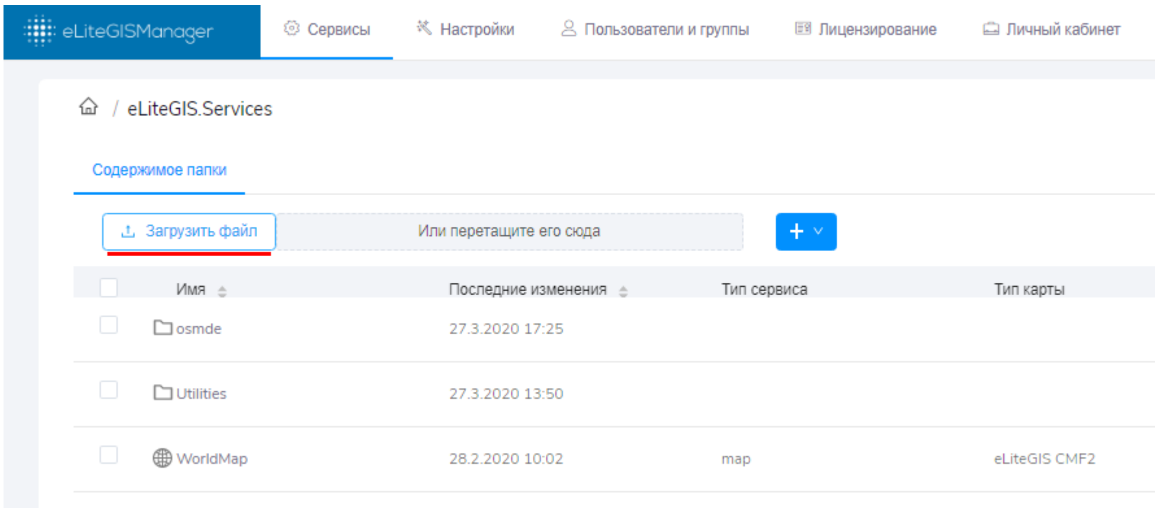
Figure 139 – Saving QGS project

Preparation of data and project for publishing geocoding service is completed.

## 7.3. Publishing the geocoding service in eLiteGIS

This section describes the steps of publishing geocoding service in **eLiteGIS**. More details about work with services (including the geocoding service) are provided in **Publishing GIS services in eLiteGIS** manual.

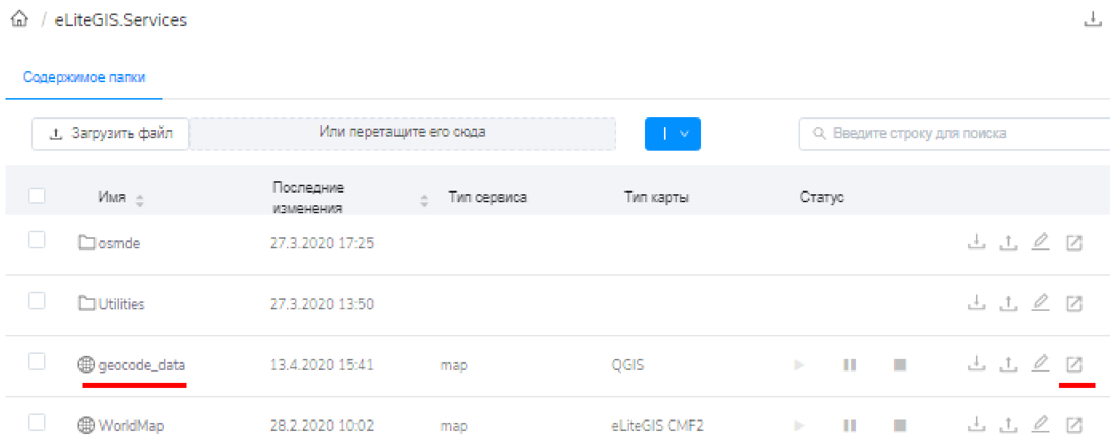
For publishing geocoding service created based on the map project (see above), open **eLiteGIS Manager** and download QGS project, see Figure 140.



| Имя | Последние изменения | Тип сервиса | Тип карты |
|---|---|---|---|
| osmde | 27.3.2020 17:25 | | |
| Utilities | 27.3.2020 13:50 | | |
| WorldMap | 28.2.2020 10:02 | map | eLiteGIS CMF2 |

Figure 140 – Downloading QGS project in eLiteGIS

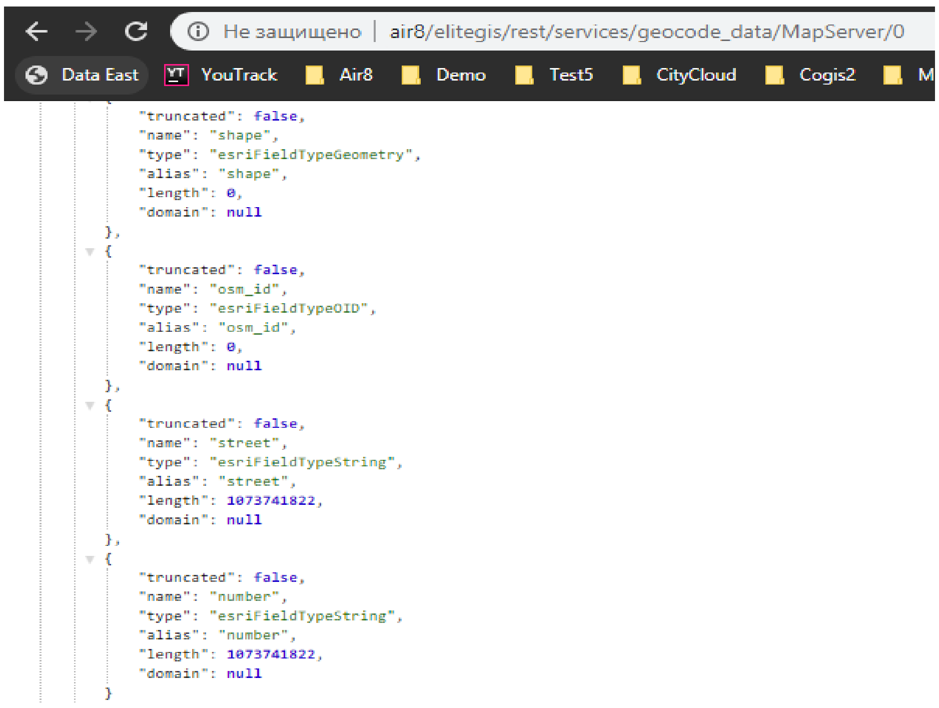
After the project download, the map service is automatically published. Check its working capability as shown on Figure 141.



| Имя | Последние изменения | Тип сервиса | Тип карты | Статус |
|---|---|---|---|---|
| osmde | 27.3.2020 17:25 | | | |
| Utilities | 27.3.2020 13:50 | | | |
| geocode_data | 13.4.2020 15:41 | map | QGIS | |
| WorldMap | 28.2.2020 10:02 | map | eLiteGIS CMF2 | |

Figure 141 – Checking the geocoding service working capability

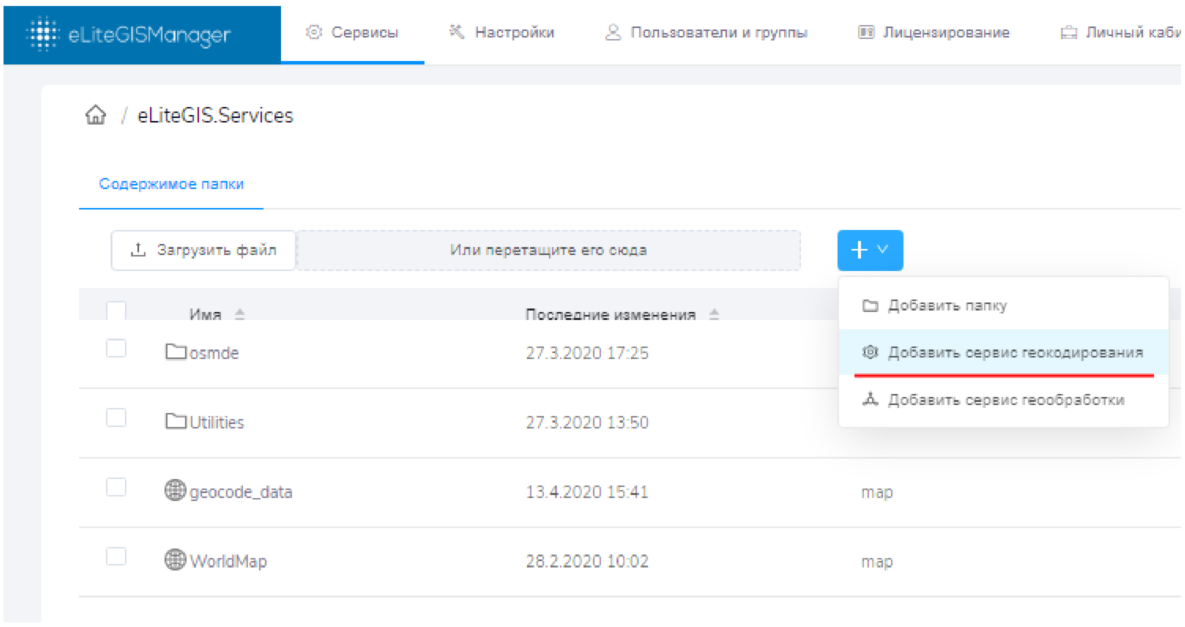
The order of fields in the service declaration should be the same as for the freetext index, see Figure 142.


Не защищено | air8/elitegis/rest/services/geocode_data/MapServer/0
Data East
YouTrack
Air8
Demo
Test5
CityCloud
Cogis2
M

```
        "truncated": false,
        "name": "shape",
        "type": "esriFieldTypeGeometry",
        "alias": "shape",
        "length": 0,
        "domain": null
    },
    {
        "truncated": false,
        "name": "osm_id",
        "type": "esriFieldTypeOID",
        "alias": "osm_id",
        "length": 0,
        "domain": null
    },
    {
        "truncated": false,
        "name": "street",
        "type": "esriFieldTypeString",
        "alias": "street",
        "length": 1073741822,
        "domain": null
    },
    {
        "truncated": false,
        "name": "number",
        "type": "esriFieldTypeString",
        "alias": "number",
        "length": 1073741822,
        "domain": null
    }
```

Figure 142 – Checking the fields order in the service declaration

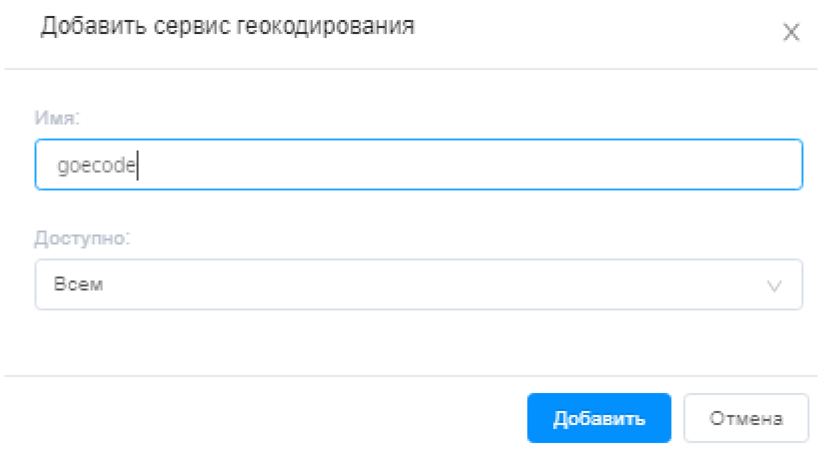
After checking completion, the geocoding service can be published, see Figure 143.



| | Имя | Последние изменения | |
|---|---|---|---|
| | osmde | 27.3.2020 17:25 | |
| | Utilities | 27.3.2020 13:50 | |
| | geocode_data | 13.4.2020 15:41 | map |
| | WorldMap | 28.2.2020 10:02 | map |

Figure 143 – Adding the geocoding service (1)

In the appeared window specify the service name and press Add button, see Figure 144.

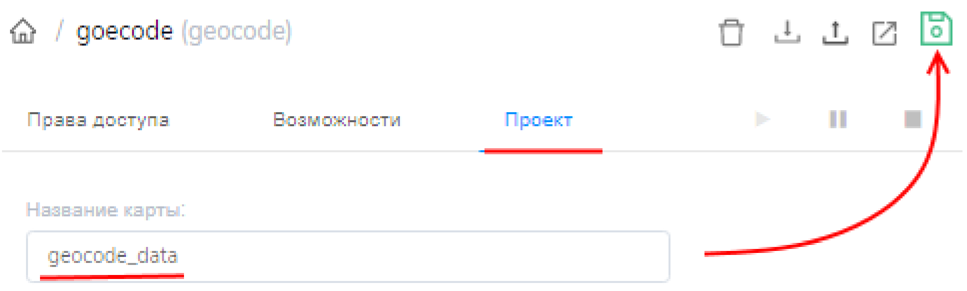

Добавить сервис геокодирования

Имя: goecode

Доступно: Всем

Figure 144 – Adding the geocoding service (2)

Then, in the Project tab of the service properties in the Map name field, select the map service that was published on the previous step based on the QGIS project. Press Save, see Figure 145.



Название карты: geocode_data

Figure 145 – Setting the geocoding service

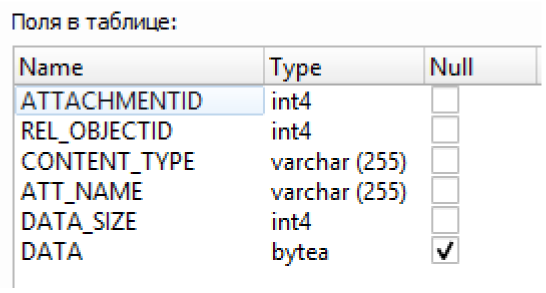
Geocoding service is ready for work.

# 8. Attachments

**eLiteGIS** supports storing photos, documents and other files as attachments to features. Setting up the storage of attachments is not done by means of QGIS, but via creation and setting of specific tables in the database.

## 8.1. Storing attachments in the database

To store attachments in the database, separate tables are created for each feature class to which you want to add attachments.
Table name by default: *<mytable>_ATTACH.*
The table can be created by means of QGIS (in the Database manager).
The table structure (field names and types) looks as following, see Figure 146.



Поля в таблице:

| Name | Type | Null |
|---|---|---|
| ATTACHMENTID | int4 | ☐ |
| REL_OBJECTID | int4 | ☐ |
| CONTENT_TYPE | varchar (255) | ☐ |
| ATT_NAME | varchar (255) | ☐ |
| DATA_SIZE | int4 | ☐ |
| DATA | bytea | ✓ |

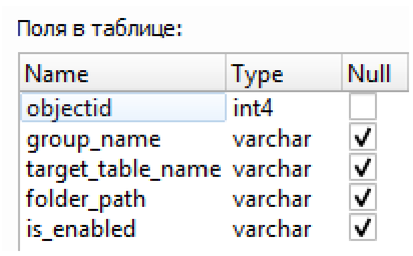
Figure 146 – Structure of table for storage of attachments in the database

The script for table creation in PostgreSQL:

```
CREATE TABLE <my_schema>."<my_table>__ATTACH"
(
  "ATTACHMENTID" serial,
  "REL_OBJECTID" integer NOT NULL,
  "CONTENT_TYPE" character varying(255) NOT NULL,
  "ATT_NAME" character varying(255) NOT NULL,
  "DATA_SIZE" integer NOT NULL,
  "DATA" bytea,
  CONSTRAINT <my_schema>."<my_table>__ATTACH" PRIMARY KEY ("ATTACHMENTID")
)
TABLESPACE pg_default;
ALTER TABLE <my_schema>."<my_table>__ATTACH"
  OWNER to postgres;
```

## 8.2. Storing attachments as files on the disk

For storage of attachments on the disk, a separate table is created, in which it is specified where the files will be saved and for which feature classes.
Table name by default: *eLiteGIS_attachment_groups*.
The table can be created by means of QGIS (in the Database manager).
The table structure (field names and types) looks as following, see Figure 147.


Поля в таблице:

| Name | Type | Null |
|---|---|---|
| objectid | int4 | |
| group_name | varchar | ✓ |
| target_table_name | varchar | ✓ |
| folder_path | varchar | ✓ |
| is_enabled | varchar | ✓ |

Figure 147 – Structure of table for storage of attachments as files on the disk

The table:

- *group_name* – the name of table record for orientation (can be not unique);
- *target_table_name* – the short of full name or template of the feature class name;
- *folder_path* – the path to the root folder where to the files should be saved;
- *is_enabled* – the property indicating whether the setting of this record should be used (0 - no, 1 - yes).

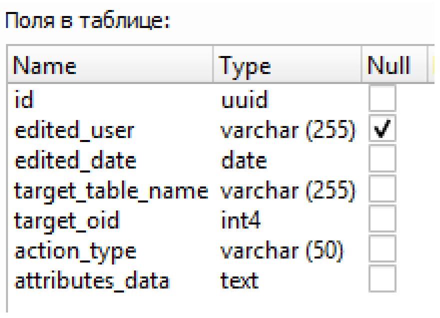
The script for table creation in PostgreSQL:

```
CREATE TABLE <my_schema>.eLiteGIS_attachment_groups
(
  objectid serial,
  group_name character varying,
  target_table_name character varying,
  folder_path character varying,
  is_enabled character varying,
  CONSTRAINT eLiteGIS_attachment_groups_pkey PRIMARY KEY (objectid)
)
TABLESPACE pg_default;
ALTER TABLE <my_schema>.eLiteGIS_attachment_groups
  OWNER to postgres;
```

## 9. Edits history

**eLiteGIS** allows you to enable recording of all edits of the features.
This setting is not done by means of QGIS, but via creation and setting of specific tables in the database.

For work of the edits history setting, the corresponding SOE rule (see details in the **Creating map applications in CoGIS** manual) and the specific table in the database are required.
Table name by default: *eLiteGIS_edit_history*
The table can be created by means of QGIS (in the Database manager).
The table structure (field names and types) looks as following, see Figure 148:



Поля в таблице:

| Name | Type | Null |
|---|---|---|
| id | uuid | ☐ |
| edited_user | varchar (255) | ✓ |
| edited_date | date | ☐ |
| target_table_name | varchar (255) | ☐ |
| target_oid | int4 | ☐ |
| action_type | varchar (50) | ☐ |
| attributes_data | text | ☐ |

Figure 148 – Structure of fields in the table for storing the edits history

The script for table creation in PostgreSQL:

```
CREATE TABLE <my_schema>.eLiteGIS_edit_history

(
  id uuid NOT NULL,
  edited_user character varying(255),
  edited_date date NOT NULL,
  target_table_name character varying(255) NOT NULL,
  target_oid integer NOT NULL,
  action_type character varying(50) NOT NULL,
  attributes_data text NOT NULL,
  CONSTRAINT eLiteGIS_edit_history_pkey PRIMARY KEY (id)
)
TABLESPACE pg_default;
ALTER TABLE <my_schema>.eLiteGIS_edit_history
  OWNER to postgres;
```

## 10. Tiles auto update

The map service, in addition to vector graphics, can output the map areas as the raster image.

Such image is divided into blocks/squares called tiles.

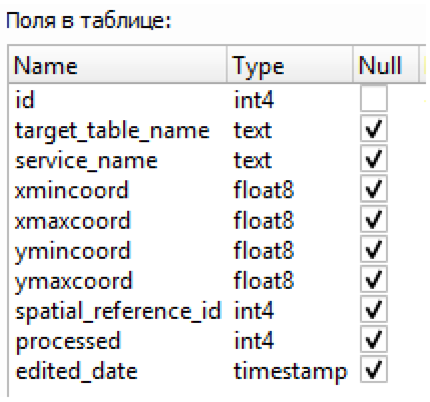
After generation, the tiles are stored in the tile cache, and on subsequent requests, the tiles are not re-generated, but taken from the cache. Thus, if map features have been changed, then these changes will not be reflected on previously generated tiles.

To avoid such situations, the map service in **eLiteGIS** can be configured to update those tiles that contain changed features. Besides the corresponding setting of service in **eLiteGIS Manager** (see **Publishing GIS services in eLiteGIS** manual), it is required to create a specific table in the database, where to the extents for tiles re-generation will be recorded.

Table name by default: *eLiteGIS_changed_extent_log.*

The table can be created by means of QGIS (in the Database manager).

The table structure (field names and types) looks as following:



Поля в таблице:

| Name | Type | Null |
|---|---|---|
| id | int4 | |
| target_table_name | text | ✓ |
| service_name | text | ✓ |
| xmincoord | float8 | ✓ |
| xmaxcoord | float8 | ✓ |
| ymincoord | float8 | ✓ |
| ymaxcoord | float8 | ✓ |
| spatial_reference_id | int4 | ✓ |
| processed | int4 | ✓ |
| edited_date | timestamp | ✓ |

Figure 149 – Structure of table for storage of extents for tiles re-generation

The script for table creation in PostgreSQL:

```
CREATE TABLE <my_schema>.eLiteGIS_changed_extent_log
(
  id serial,
  target_table_name text,
  service_name text,
  xmincoord double precision,
  xmaxcoord double precision,
  ymincoord double precision,
  ymaxcoord double precision,
  spatial_reference_id integer,
  processed integer,
  edited_date timestamp with time zone,
  CONSTRAINT eLiteGIS_changed_extent_log_pkey PRIMARY KEY (id)
)
TABLESPACE pg_default;
ALTER TABLE <my_schema>.eLiteGIS_changed_extent_log
  OWNER to postgres;
```

Also, to fill this table in the database, you should create the trigger for the change in the corresponding feature class. If there are several feature classes, then the trigger is created for each such class.
The trigger creation script is as following:

```
CREATE FUNCTION <my_schema>.set_data_to_extentchangelog() RETURNS trigger AS $$
   BEGIN
      IF (OLD.geom IS NOT NULL AND ST_IsEmpty(OLD.geom) = false) THEN
         BEGIN
            INSERT INTO <my_schema>.eLiteGIS_changed_extent_log (edited_date,
target_table_name, xmincoord, xmaxcoord, ymincoord, ymaxcoord,
spatial_reference_id)
            VALUES (CURRENT_TIMESTAMP,
      concat(TG_TABLE_SCHEMA, '.', TG_TABLE_NAME),
                  ST_XMin(OLD.geom),
                  ST_XMax(OLD.geom),
                  ST_YMin(OLD.geom),
                  ST_YMax(OLD.geom),
                  ST_SRID(OLD.geom));
         END;
      END IF;

      IF (NEW.geom IS NOT NULL AND ST_IsEmpty(NEW.geom) = false) THEN
         BEGIN
            INSERT INTO <my_schema>.eLiteGIS_changed_extent_log (edited_date,
target_table_name, xmincoord, xmaxcoord, ymincoord, ymaxcoord,
spatial_reference_id)
            VALUES (CURRENT_TIMESTAMP,
                  concat(TG_TABLE_SCHEMA, '.', TG_TABLE_NAME),
                  ST_XMin(NEW.geom),
                  ST_XMax(NEW.geom),
                  ST_YMin(NEW.geom),
                  ST_YMax(NEW.geom),
                  ST_SRID(NEW.geom));
         END;
      END IF;

      RETURN NULL;
   END;
$$ LANGUAGE plpgsql;

CREATE TRIGGER set_data_to_extent_change_log
AFTER INSERT OR UPDATE OR DELETE
ON '<my_schema>.<my_featureclass>
         FOR EACH ROW
```

```
EXECUTE PROCEDURE <my_schema>.eLiteGIS_changed_extent_log();
```