CoGIS Server
Creating map projects in QGIS

Setting parameters of map projects for further publication in CoGIS Server

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;
- **CoGIS Server** – a GIS server for publishing data and tools as web services.

CoGIS Server components

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

CoGIS Server 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 CoGIS Server.

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 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:

- CoGIS Server - Publishing GIS services;
- CoGIS Server - Installing and setting;
- CoGIS Portal - Installing and setting;
- CoGIS Server - Creating map projects in QGIS;
- CoGIS Portal - Creating map applications;

- CoGIS Mobile - Working in mobile applications.

2. Settings of connection to database

2.1. Supported DBMS

CoGIS Server 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.

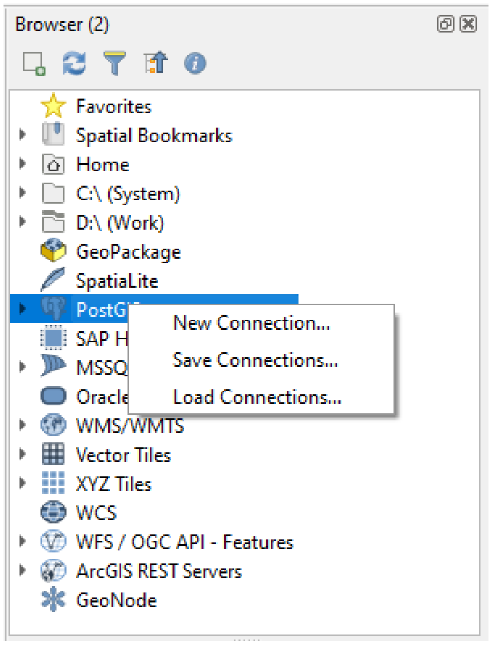
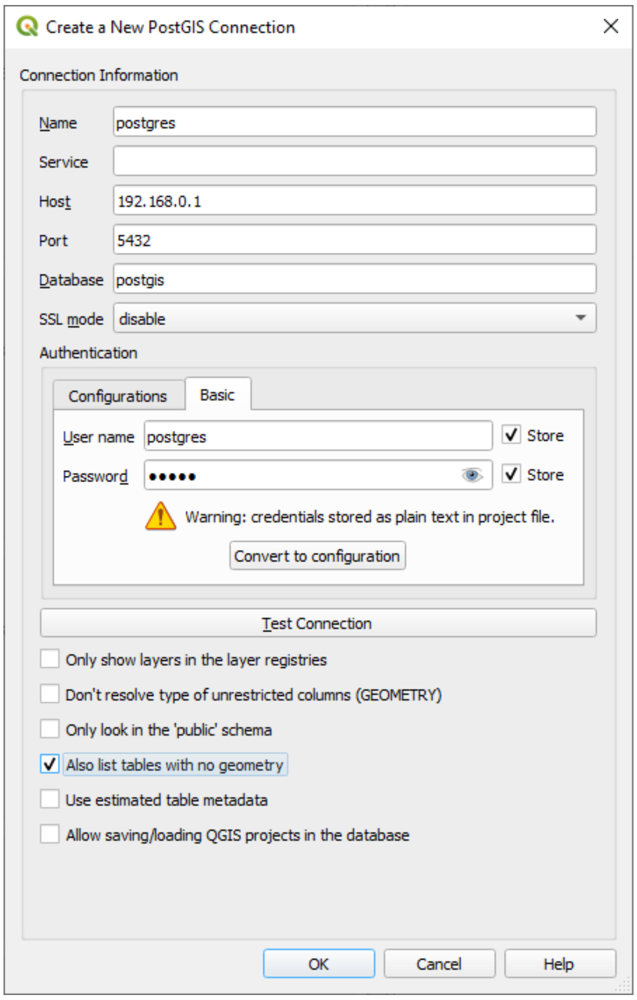


Figure 1 – Connection to database

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


Create a New PostGIS Connection

Connection Information

Name: postgres
Service:
Host: 192.168.0.1
Port: 5432
Database: postgis
SSL mode: disable

Authentication

Configurations | Basic

User name: postgres [✓ Store]
Password: ••••• [✓ Store]

Warning: credentials stored as plain text in project file.

Convert to configuration

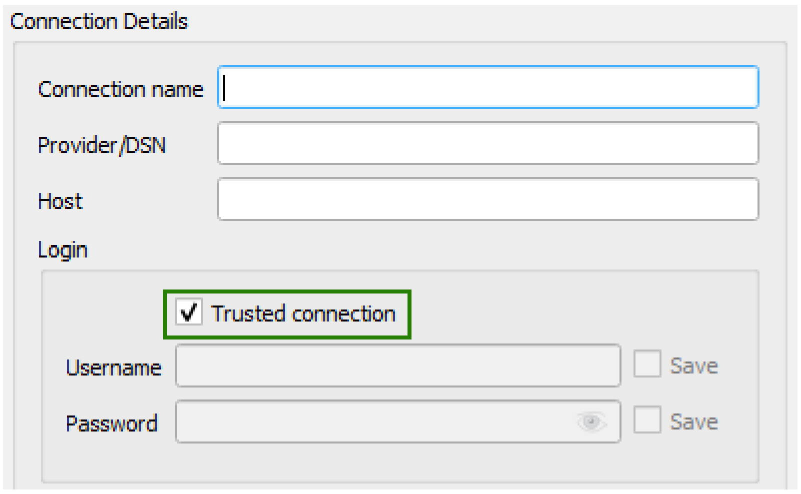
Test Connection

- [ ] Only show layers in the layer registries
- [ ] Don't resolve type of unrestricted columns (GEOMETRY)
- [ ] Only look in the 'public' schema
- [x] Also list tables with no geometry
- [ ] Use estimated table metadata
- [ ] Allow saving/loading QGIS projects in the database

OK | Cancel | Help

Figure 2 – Entering connection parameters

For connection to MSSQL, the possibility of connection via the end-to-end Windows authorization without entering login-password may be relevant. In QGIS, in the Connection to database settings there is the “trust connect” setting, see Figure 3 **– Trust connect setting for database** , which means connection under the current Windows user under which the process is running. In terms of CoGIS Server this is the user under which AppPool is running (the CoGIS Server process is running).


Connection Details

Connection name:
Provider/DSN:
Host:

Login

- [x] Trusted connection

Username: [ ] Save
Password: [ ] Save

Figure 3 – Trust connect setting for database

If this option is enabled, the Windows authorization will be used for connection to the database.

2.3. Database manager

Working with the connected database in QGIS is recommended via Database manager, see Figure 4.

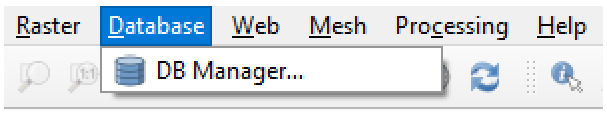


Figure 4 – Database manager in QGIS

2.3.1. Creating table

To create the table in the database, select the appropriate menu option in Database manager, see Figure 5.

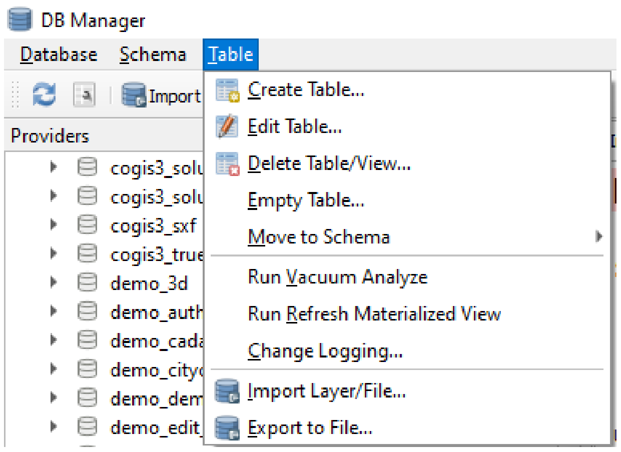
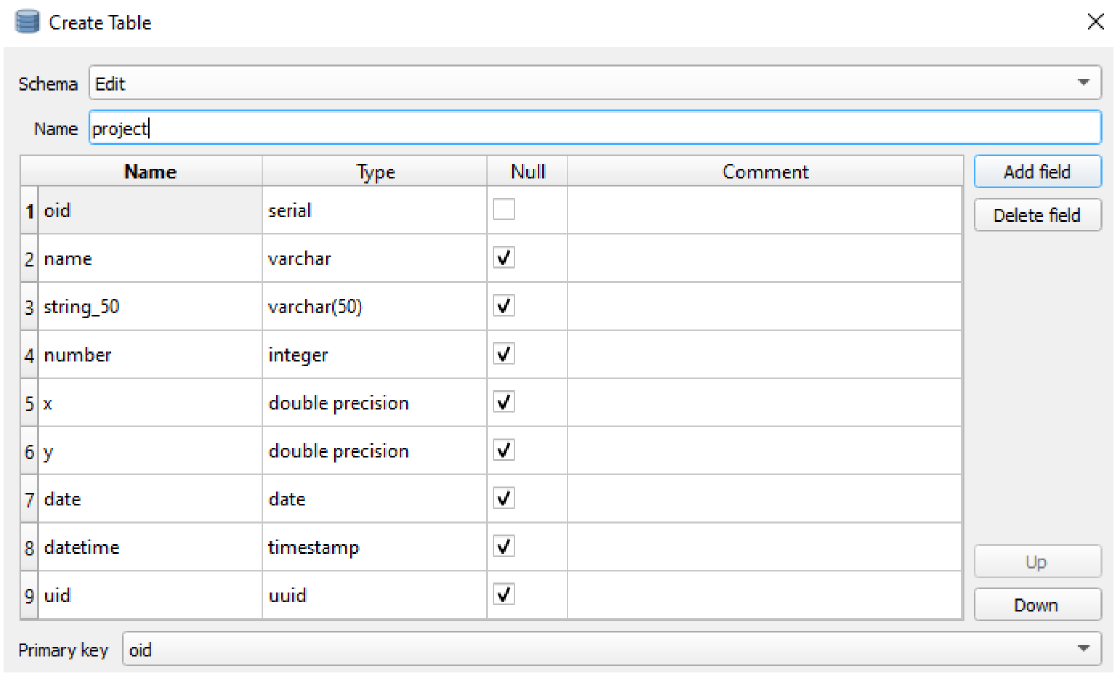


Figure 5 – Creation of new table in database

2.3.2. General types of fields in the table

The list of general types of fields in the table is shown on Figure 6.



Schema: Edit

Name: project

	Name	Type	Null	Comment
1	oid	serial	☐	
2	name	varchar	✓	
3	string_50	varchar(50)	✓	
4	number	integer	✓	
5	x	double precision	✓	
6	y	double precision	✓	
7	date	date	✓	
8	datetime	timestamp	✓	
9	uid	uuid	✓	

Primary key: oid

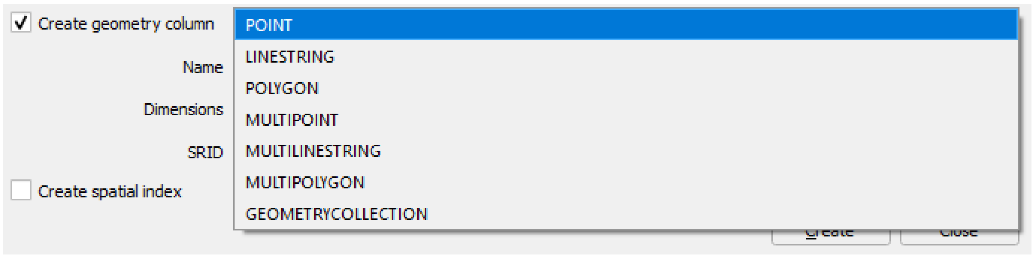
Figure 6 – General types of fields in the table

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 7.



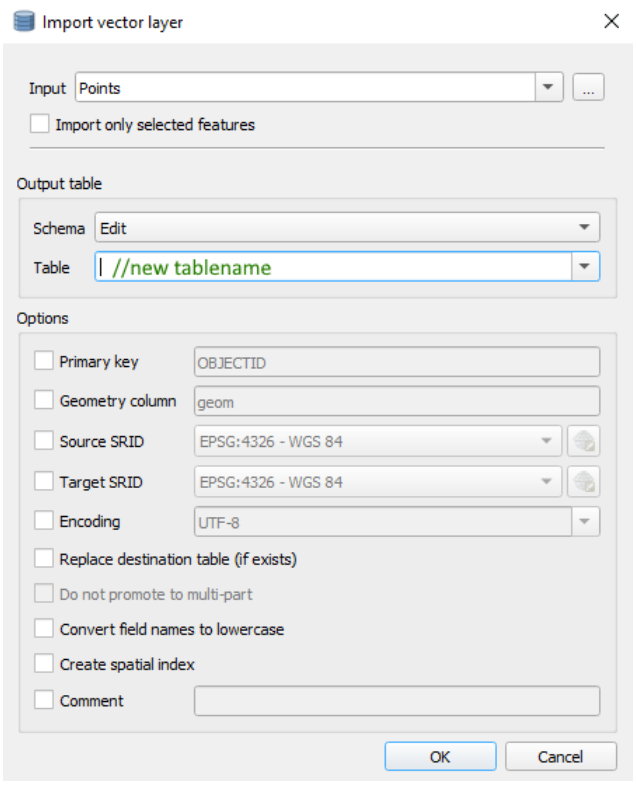
☑ Create geometry column
Name
Dimensions
SRID
☐ Create spatial index

POINT
LINESTRING
POLYGON
MULTIPOINT
MULTILINESTRING
MULTIPOLYGON
GEOMETRYCOLLECTION

Figure 7 – 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 8.



Import vector layer

Input: Points
☐ Import only selected features

Output table
Schema: Edit
Table: //new tablename

Options
☐ Primary key: OBJECTID
☐ Geometry column: geom
☐ Source SRID: EPSG:4326 - WGS 84
☐ Target SRID: EPSG:4326 - WGS 84
☐ Encoding: UTF-8
☐ Replace destination table (if exists)
☐ Do not promote to multi-part
☐ Convert field names to lowercase
☐ Create spatial index
☐ Comment

OK Cancel

Figure 8 – 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 9.

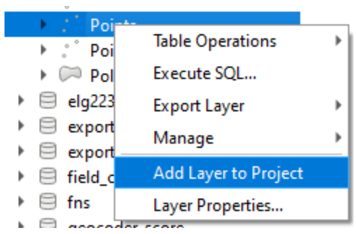
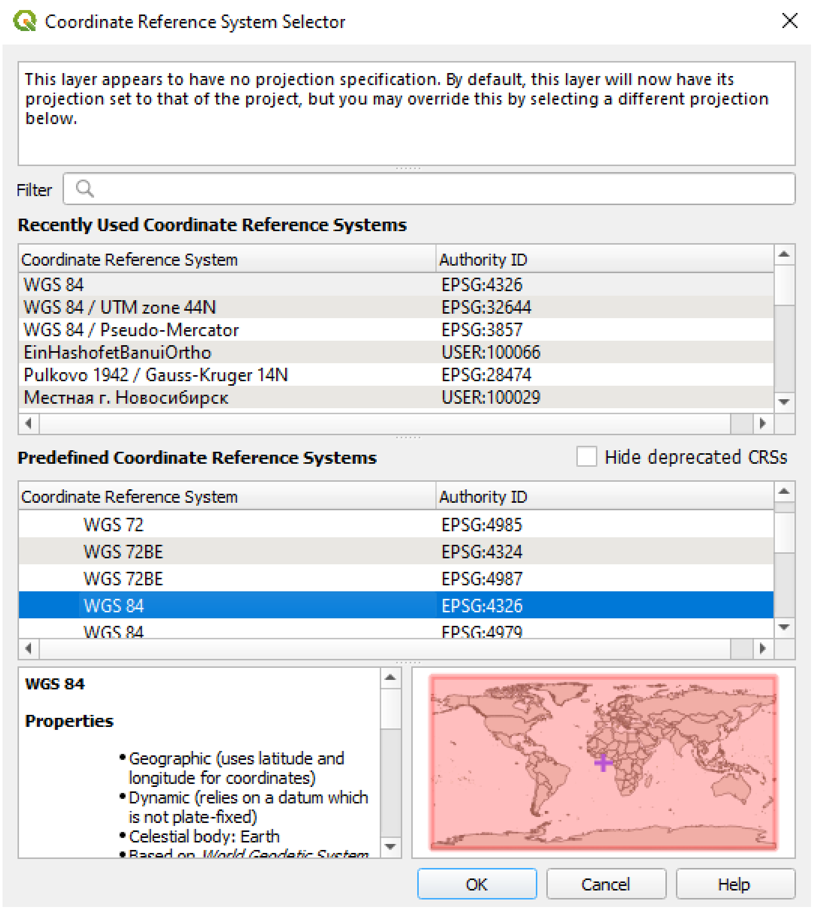


Figure 9 – Adding table to the project map

Now you need to select coordinate system for the added layer, see Figure 10.


Coordinate Reference System Selector

This layer appears to have no projection specification. By default, this layer will now have its projection set to that of the project, but you may override this by selecting a different projection below.

Filter

Recently Used Coordinate Reference Systems

Coordinate Reference System	Authority ID
WGS 84	EPSG:4326
WGS 84 / UTM zone 44N	EPSG:32644
WGS 84 / Pseudo-Mercator	EPSG:3857
EinHashofetBanuiOrtho	USER:100066
Pulkovo 1942 / Gauss-Kruger 14N	EPSG:28474
Местная г. Новосибирск	USER:100029

Predefined Coordinate Reference Systems Hide deprecated CRSs

Coordinate Reference System	Authority ID
WGS 72	EPSG:4985
WGS 72BE	EPSG:4324
WGS 72BE	EPSG:4987
WGS 84	EPSG:4326
WGS 84	EPSG:4979

WGS 84

Properties

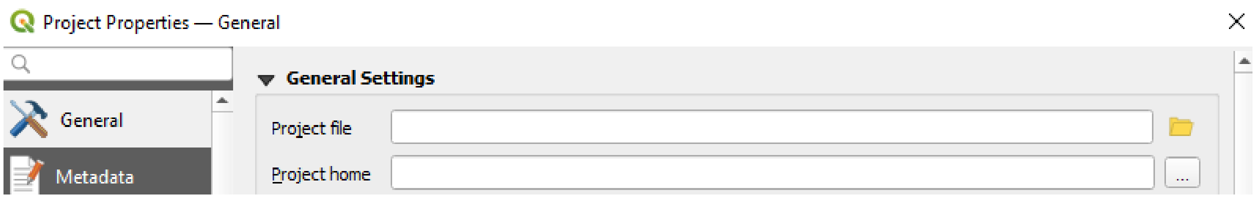
- Geographic (uses latitude and longitude for coordinates)
- Dynamic (relies on a datum which is not plate-fixed)
- Celestial body: Earth
- Based on *World Geodetic System*

Figure 10 – 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 11.

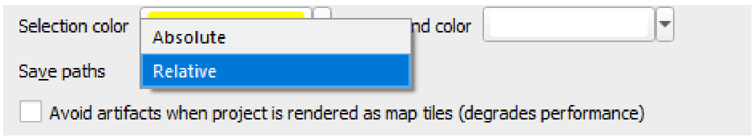


Project file:

Project home:

Figure 11 – General project's properties

Now you need to set the saved paths to the data sources as shown on Figure 12.



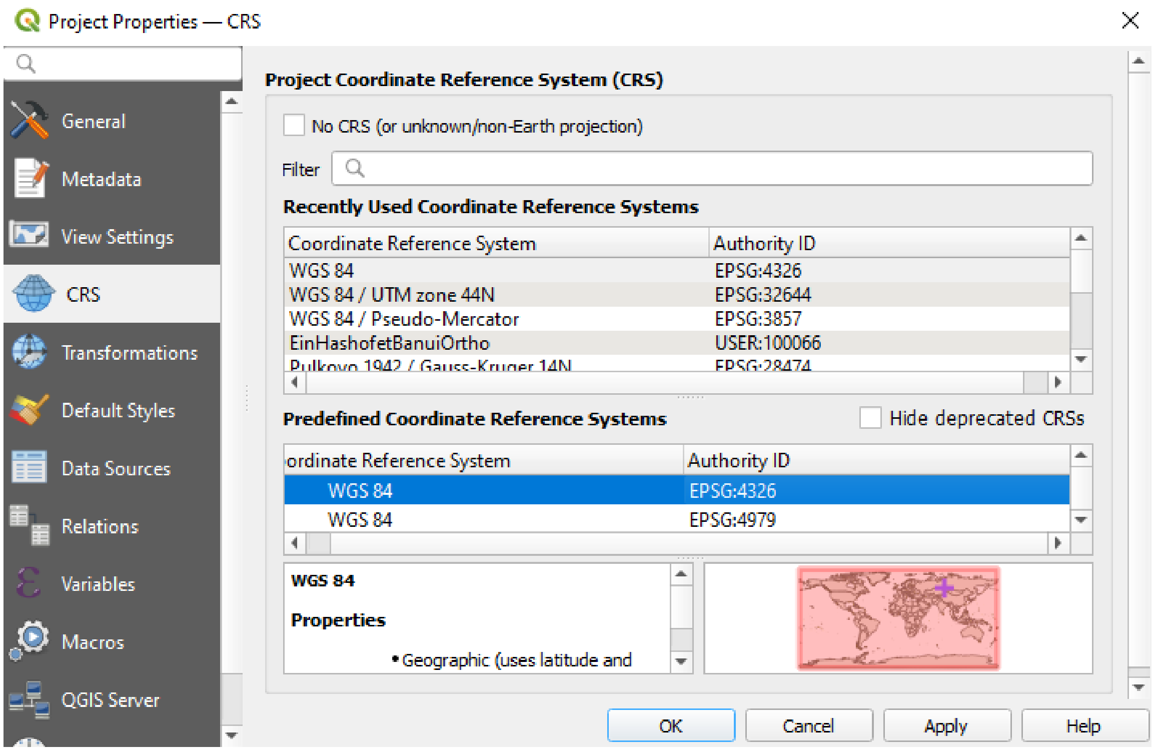
Save paths: Absolute [✓ Relative]

☐ Avoid artifacts when project is rendered as map tiles (degrades performance)

Figure 12 – Paths to data sources

3.2. Coordinate system

For the QGIS project you need to set the coordinate system, see Figure 13.



Project Coordinate Reference System (CRS)

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

Filter

Recently Used Coordinate Reference Systems

Coordinate Reference System	Authority ID
WGS 84	EPSG:4326
WGS 84 / UTM zone 44N	EPSG:32644
WGS 84 / Pseudo-Mercator	EPSG:3857
EinHashofetBanuiOrtho	USER:100066
Pulkovo 1942 / Gauss-Kruger 14N	EPSG:28474

Predefined Coordinate Reference Systems ☐ Hide deprecated CRSs

ordinate Reference System	Authority ID
WGS 84	EPSG:4326
WGS 84	EPSG:4979

WGS 84

Properties

• Geographic (uses latitude and

Figure 13 – Setting coordinate system for the project

The *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 14.

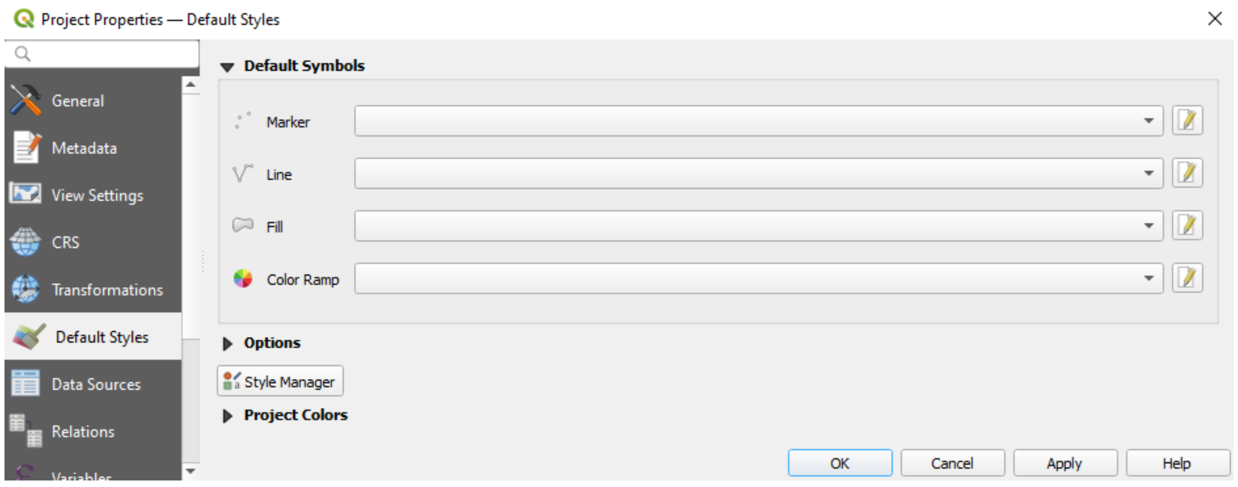
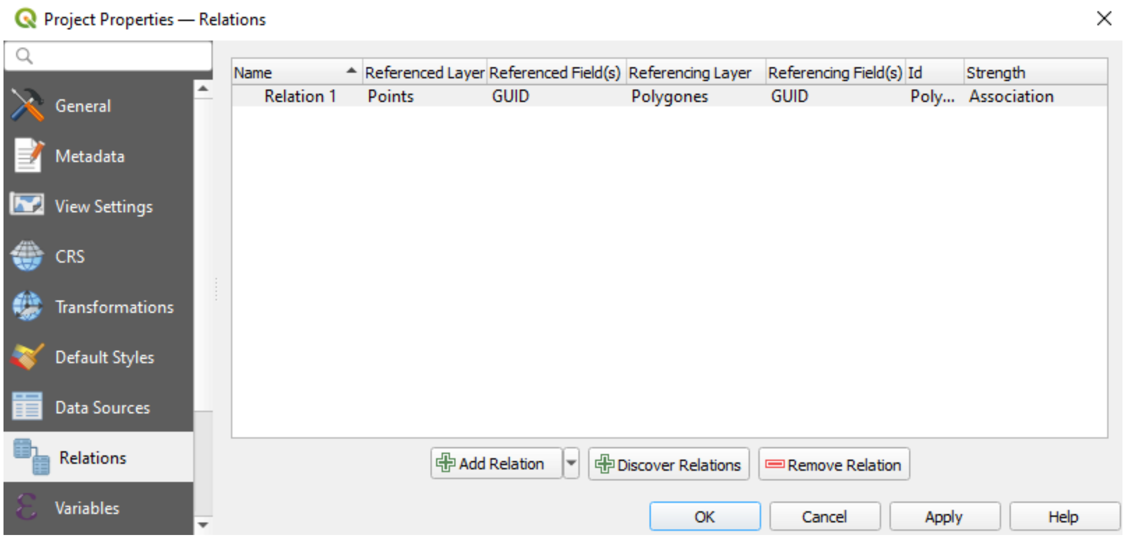


Figure 14 – Default styles for the project

3.4. Relations

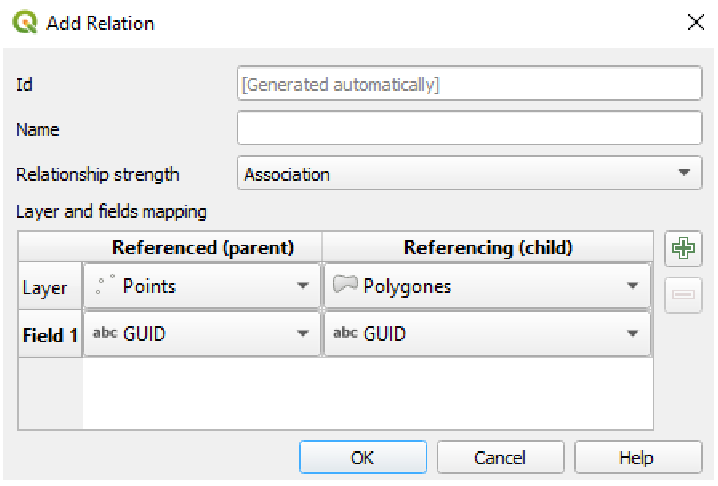
At the QGIS project level it is possible to set relations between layers and tables, that will be supported by publishing services in **CoGIS Server**, and then by building the map in **CoGIS Portal**. At that time no additional settings at the **CoGIS Server** or **CoGIS Portal** levels will be required.

The relations between layers or tables should be set based on the values from the common attribute field, see Figure 15 and Figure 16.



Name	Referenced Layer	Referenced Field(s)	Referencing Layer	Referencing Field(s)	Id	Strength
Relation 1	Points	GUID	Polygones	GUID	Poly...	Association

Figure 15 – Relations between project's layers/tables



Id: [Generated automatically]

Name:

Relationship strength: Association

Layer and fields mapping

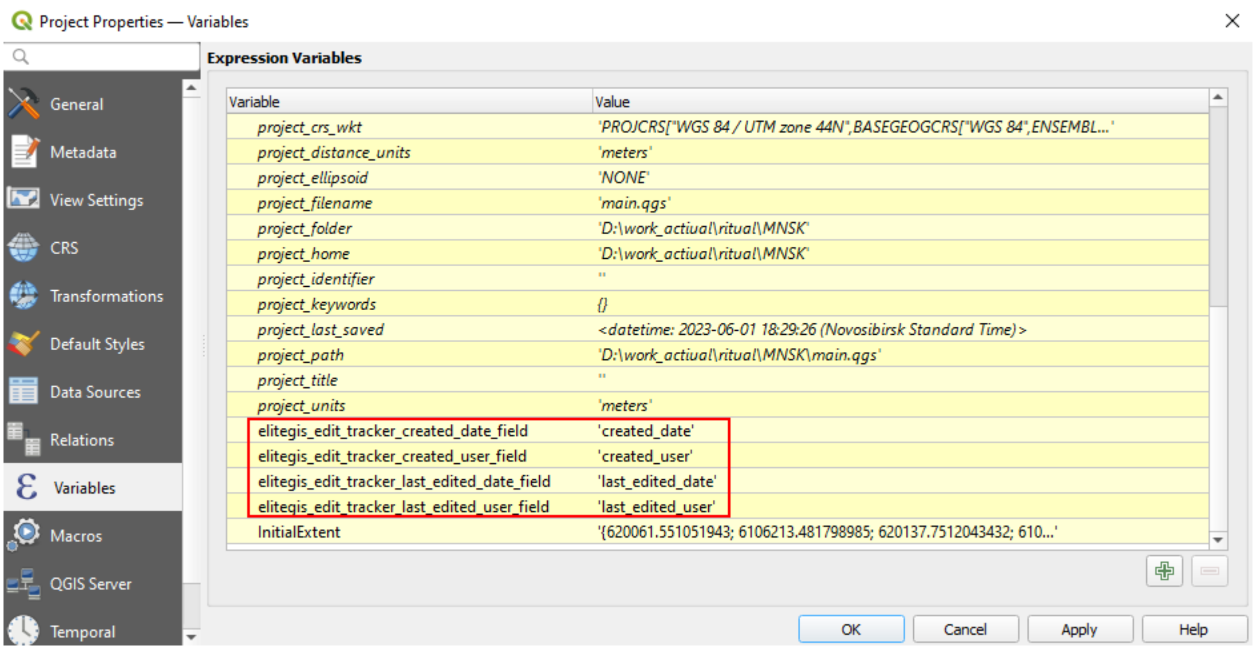
	Referenced (parent)	Referencing (child)
Layer	Points	Polygones
Field 1	abc GUID	abc GUID

Figure 16 – 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 **CoGIS Server**, 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 17.



Expression Variables

Variable	Value
project_crs_wkt	*'PROJCRS["WGS 84 / UTM zone 44N",BASEGEOGCRS["WGS 84",ENSEMBL...'*
project_distance_units	*'meters'*
project_ellipsoid	*'NONE'*
project_filename	*'main.qgs'*
project_folder	*'D:\work_actiual\ritual\MNSK'*
project_home	*'D:\work_actiual\ritual\MNSK'*
project_identifier	*''*
project_keywords	*{}*
project_last_saved	*<datetime: 2023-06-01 18:29:26 (Novosibirsk Standard Time)>*
project_path	*'D:\work_actiual\ritual\MNSK\main.qgs'*
project_title	*''*
project_units	*'meters'*
elitegis_edit_tracker_created_date_field	'created_date'
elitegis_edit_tracker_created_user_field	'created_user'
elitegis_edit_tracker_last_edited_date_field	'last_edited_date'
elitegis_edit_tracker_last_edited_user_field	'last_edited_user'
InitialExtent	'{620061.551051943; 6106213.481798985; 620137.7512043432; 610...'

Figure 17 – 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 18.

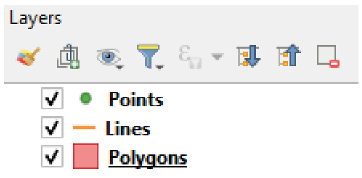


Figure 18 – Feature layers of different types

4.2. Raster layers

The raster layers are layers containing raster data only. It is also possible to publish the raster layers based on the GeoTIFF file directly via the **CoGIS Server** interface (web console **CoGIS Server Manager**, see **CoGIS Server - Publishing GIS services**for details).

4.3. Group layer

The group layer or complex layer is the layer containing sublayers of one or multiple types, see Figure 19.

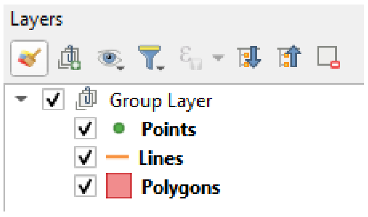
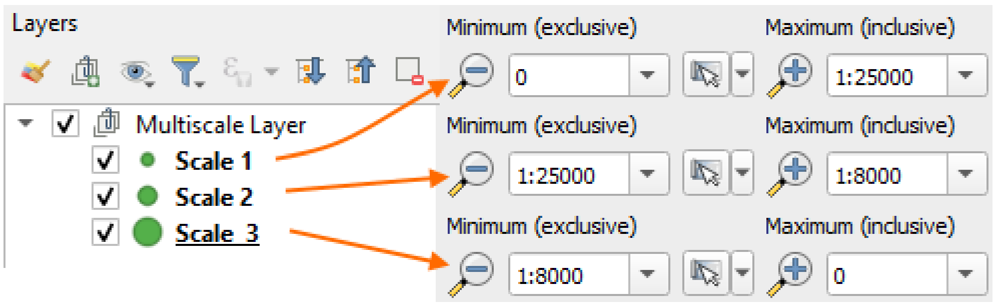


Figure 19 – Group layer

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 20.



Minimum (exclusive) 0 — Maximum (inclusive) 1:25000

Minimum (exclusive) 1:25000 — Maximum (inclusive) 1:8000

Minimum (exclusive) 1:8000 — Maximum (inclusive) 0

Figure 20 – 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 **CoGIS Server**, see Figure 21.

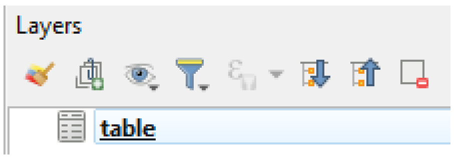


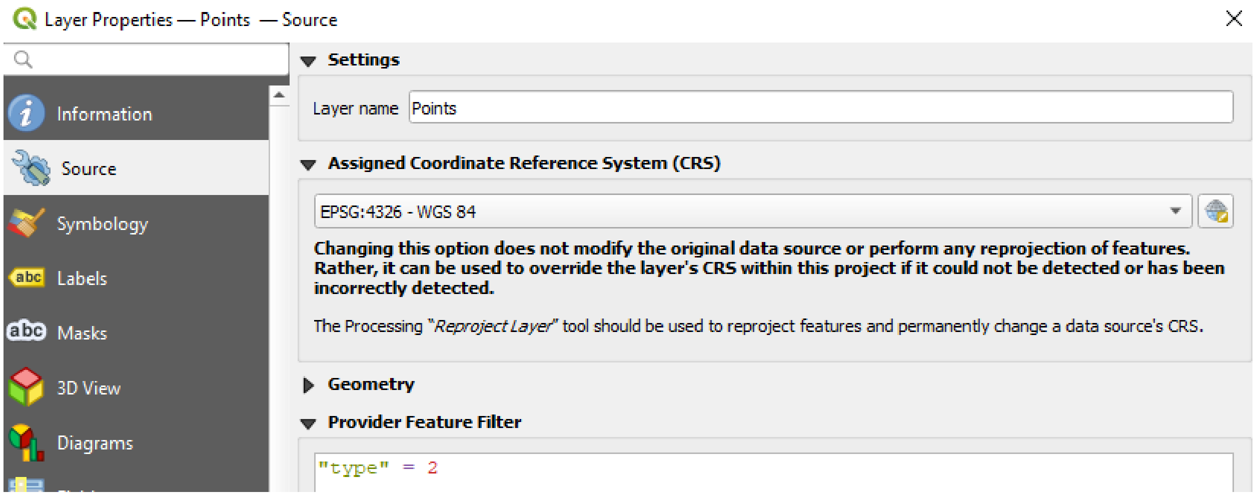
Figure 21 – The table in the list of layers of map project

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 22.



Layer name: Points

EPSG:4326 - WGS 84

Changing this option does not modify the original data source or perform any reprojection of features. Rather, it can be used to override the layer's CRS within this project if it could not be detected or has been incorrectly detected.

The Processing "*Reproject Layer*" tool should be used to reproject features and permanently change a data source's CRS.

```
"type" = 2
```

Figure 22 – 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 22.

5.1.3. Definition query

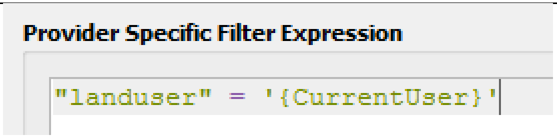
In the QGS project in the layer's properties it is possible to set the SQL definition query for specific layer: Layer's data source- *Provider Feature Filter*, see Figure 22.

5.1.4. Definition query macros

In the definition query it is possible to specify macros that will be used in the published service. The macros should be specified using single quotes, so that QGIS allows you to save the filter, see Figure 23.

Supported macros:

{CurrentUser}	Name of current user
{CurrentGroups}	Set of groups where current user is included
{CurrentGroup.mygroup}	Whether current user is included to specific group (0 – no, 1 – yes)
{CurrentDate}	Current date
{CurrentYear}	Current year
{CurrentMonth}	Current month
{CurrentDay}	Current day



```
"landuser" = '{CurrentUser}'
```

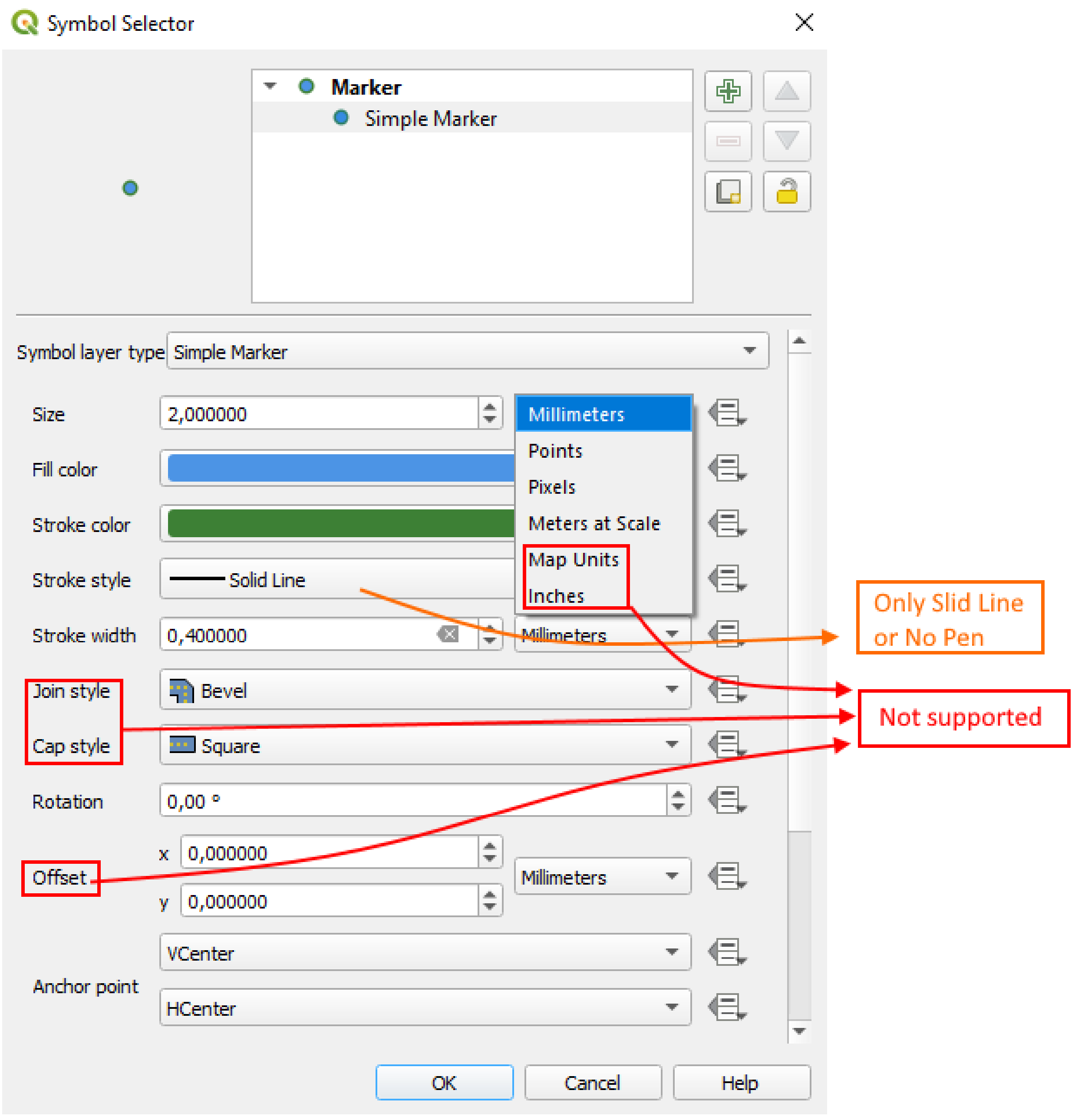
Figure 23 – Example of specifying macros in the layer filter

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 **CoGIS Server** 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 24. The figure shows which display parameters are supported by publishing the project as the map service in **CoGIS Server**.



Symbol layer type: Simple Marker
Size: 2,000000 Millimeters
Fill color
Stroke color
Stroke style: Solid Line
Stroke width: 0,400000 Millimeters
Join style: Bevel
Cap style: Square
Rotation: 0,00 °
Offset: x 0,000000 y 0,000000 Millimeters
Anchor point: VCenter HCenter

Figure 24 – 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 CoGIS Server).

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 25.

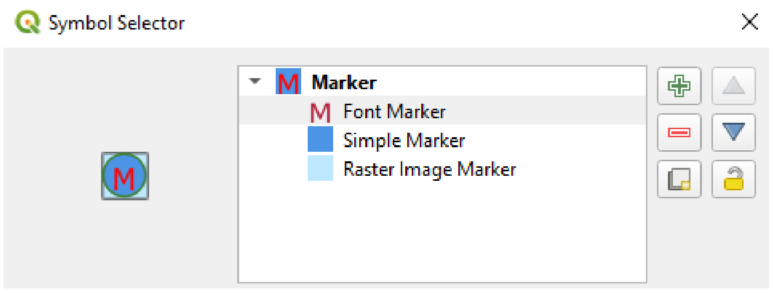
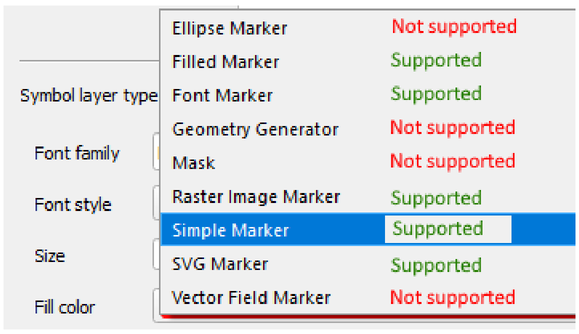


Figure 25 – Composite marker for layer

5.2.4. Types of symbols: marker

The figure below shows the types of symbols supported by publishing of map project as service in CoGIS Server, see Figure 26.



Ellipse Marker	Not supported
Filled Marker	Supported
Font Marker	Supported
Geometry Generator	Not supported
Mask	Not supported
Raster Image Marker	Supported
Simple Marker	Supported
SVG Marker	Supported
Vector Field Marker	Not supported

Figure 26 – Types of symbols supported by publishing of map project as service

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, see Figure 27.

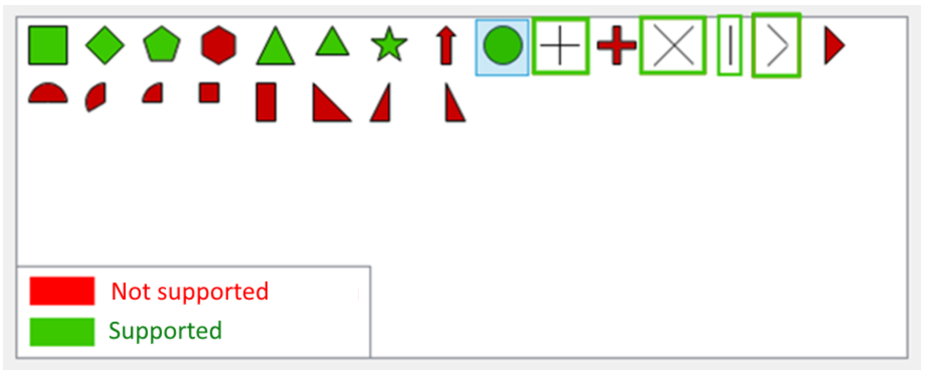


Figure 27 – Types of simple markers supported by publishing of map project as service

5.2.4.2. Symbol marker

By publishing of map project as service in **CoGIS Server**, the use of symbol markers (markers created from symbols of the main fonts from the system folder) is supported, see Figure 28.

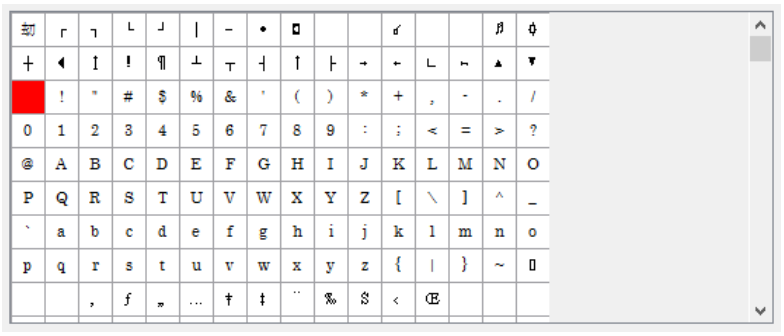


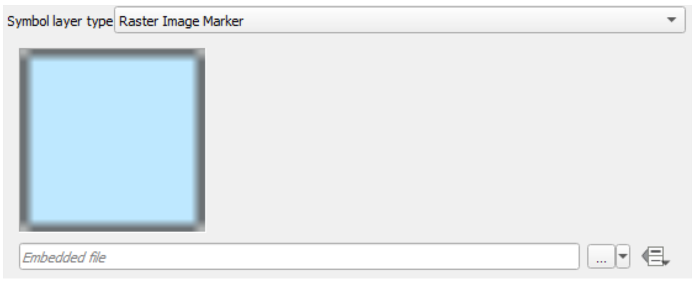
Figure 28 – Symbol marker

5.2.4.3. Fill marker

By publishing of map project as a service in **CoGIS Server**, 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 **CoGIS Server**, 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 29.



Symbol layer type: Raster Image Marker

Embedded file

Figure 29 – 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 **CoGIS Server**, the use of SVG marker created based on graphic file in SVG (*.svg) format is supported, see Figure 30.

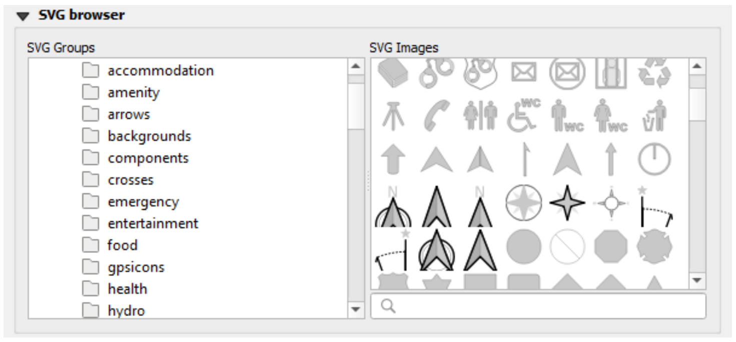
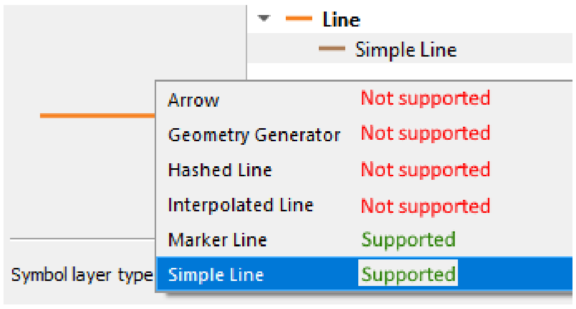


Figure 30 – SVG marker

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 **CoGIS Server**, see Figure 31.

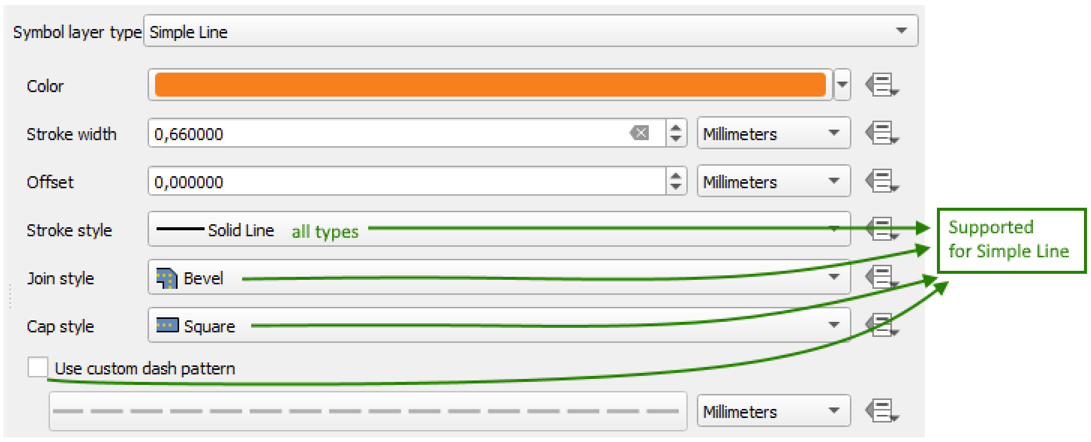


Arrow	Not supported
Geometry Generator	Not supported
Hashed Line	Not supported
Interpolated Line	Not supported
Marker Line	Supported
Simple Line	Supported

Figure 31 – Types of symbols for polylines supported by publishing of map project as service

5.2.5.1. Simple polyline

The figure below shows parameters for simple polyline supported by publishing of map project as service in **CoGIS Server**, see Figure 32.

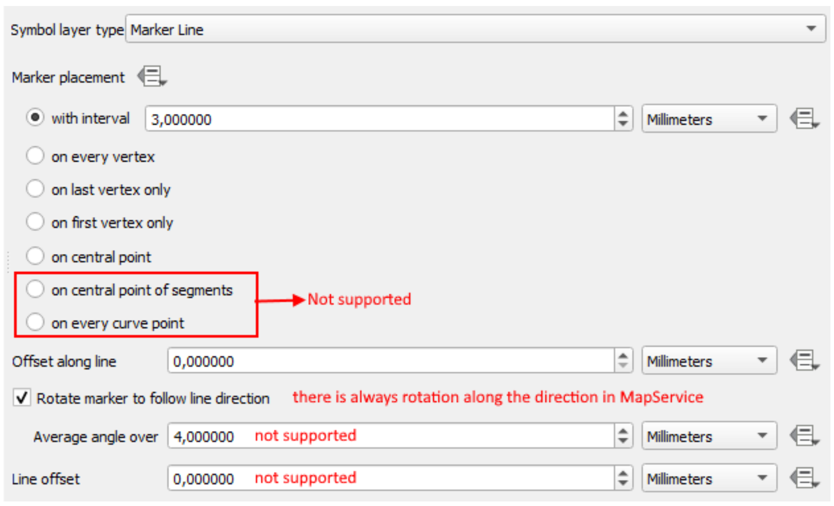


Symbol layer type: Simple Line
Color
Stroke width: 0,660000 Millimeters
Offset: 0,000000 Millimeters
Stroke style: Solid Line — all types
Join style: Bevel
Cap style: Square
Use custom dash pattern
Millimeters

Figure 32 – 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 **CoGIS Server**, see Figure 33.



Symbol layer type: Marker Line

Marker placement

- [✓ with interval] 3,000000 Millimeters
- on every vertex
- on last vertex only
- on first vertex only
- on central point
- on central point of segments
- on every curve point

on central point of segments, on every curve point → Not supported

Offset along line 0,000000 Millimeters

[✓] Rotate marker to follow line direction — there is always rotation along the direction in MapService

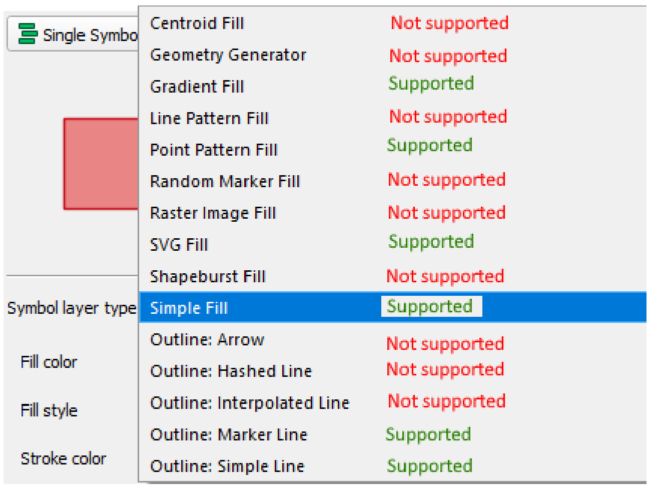
Average angle over 4,000000 Millimeters — not supported

Line offset 0,000000 Millimeters — not supported

Figure 33 – 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 **CoGIS Server**, see Figure 34.

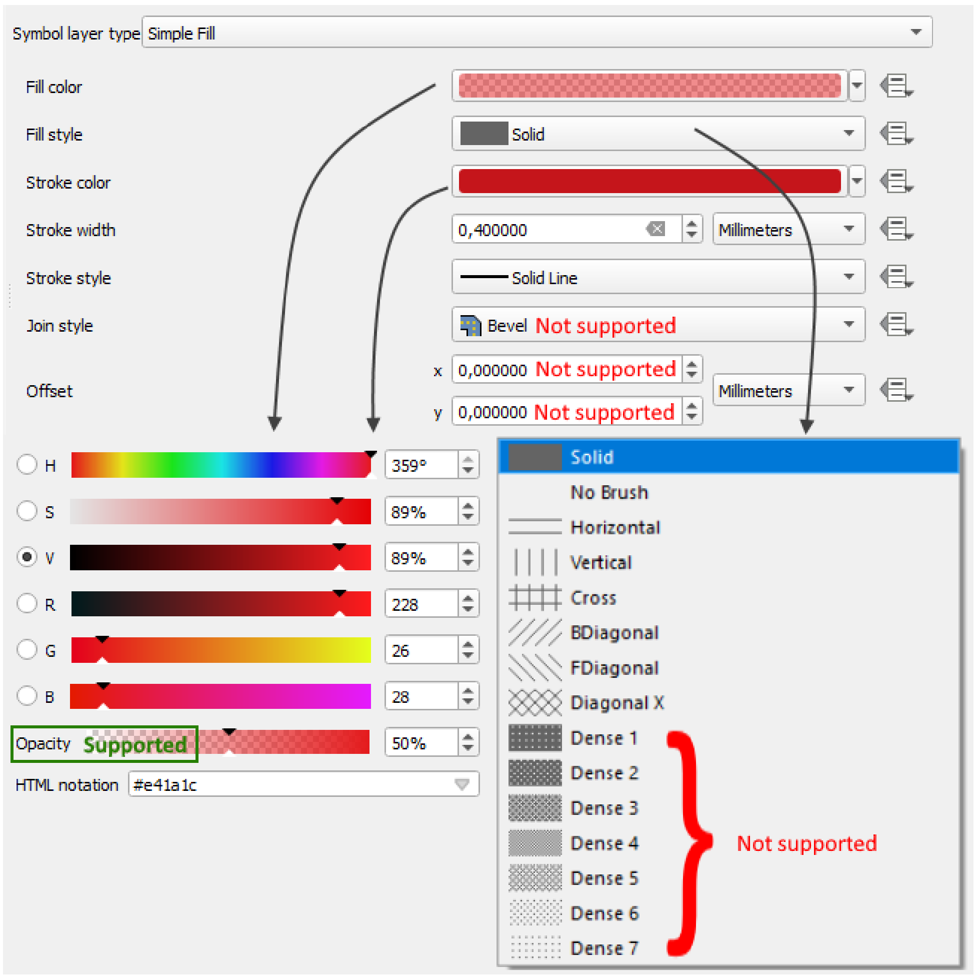


Centroid Fill	Not supported
Geometry Generator	Not supported
Gradient Fill	Supported
Line Pattern Fill	Not supported
Point Pattern Fill	Supported
Random Marker Fill	Not supported
Raster Image Fill	Not supported
SVG Fill	Supported
Shapeburst Fill	Not supported
Simple Fill	Supported
Outline: Arrow	Not supported
Outline: Hashed Line	Not supported
Outline: Interpolated Line	Not supported
Outline: Marker Line	Supported
Outline: Simple Line	Supported

Figure 34 – Types of fill supported by publishing of map project as service

5.2.6.1. Simple fill

The figure below shows parameters for simple fill supported by publishing of map project as service in **CoGIS Server**, see Figure 35.



Symbol layer type: Simple Fill

Fill color

Fill style: Solid

Stroke color

Stroke width: 0,400000 Millimeters

Stroke style: Solid Line

Join style: Bevel

Offset: x 0,000000, y 0,000000 Millimeters

H: 359°

S: 89%

V: 89%

R: 228

G: 26

B: 28

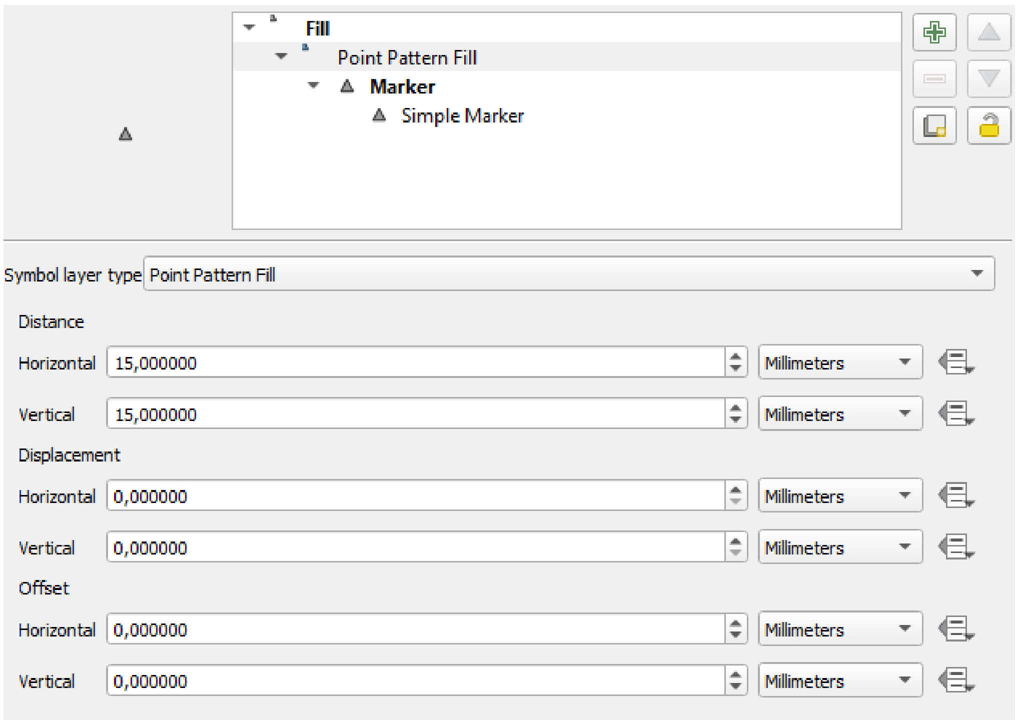
Opacity: 50%

HTML notation: #e41a1c

Figure 35 – Simple fill

5.2.6.2. Marker fill

By publishing of map project as service in **CoGIS Server**, the marker fill is supported, see Figure 36.



Fill
Point Pattern Fill
Marker
Simple Marker

Symbol layer type: Point Pattern Fill

Distance
Horizontal: 15,000000 Millimeters
Vertical: 15,000000 Millimeters

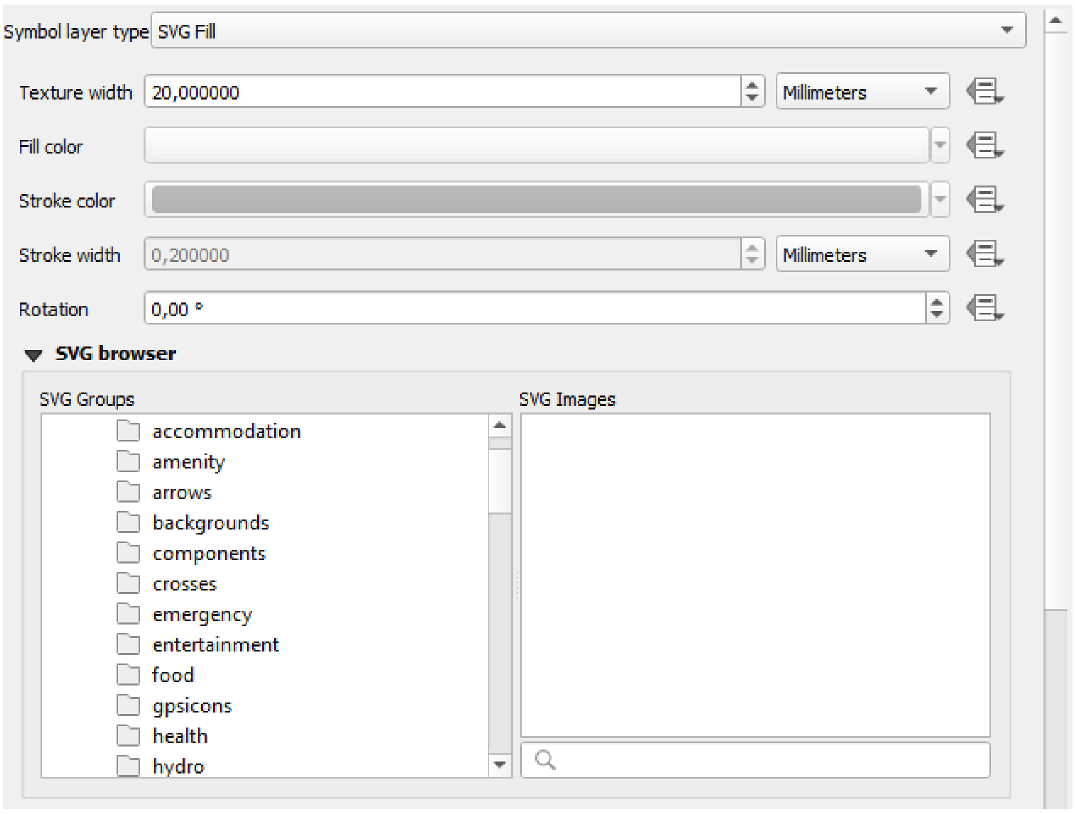
Displacement
Horizontal: 0,000000 Millimeters
Vertical: 0,000000 Millimeters

Offset
Horizontal: 0,000000 Millimeters
Vertical: 0,000000 Millimeters

Figure 36 – Marker fill

5.2.6.3. SVG template fill

By publishing of map project as service in **CoGIS Server**, the SVG template fill is supported, see Figure 37.



Symbol layer type: SVG Fill
Texture width: 20,000000 Millimeters
Fill color
Stroke color
Stroke width: 0,200000 Millimeters
Rotation: 0,00 °

SVG browser

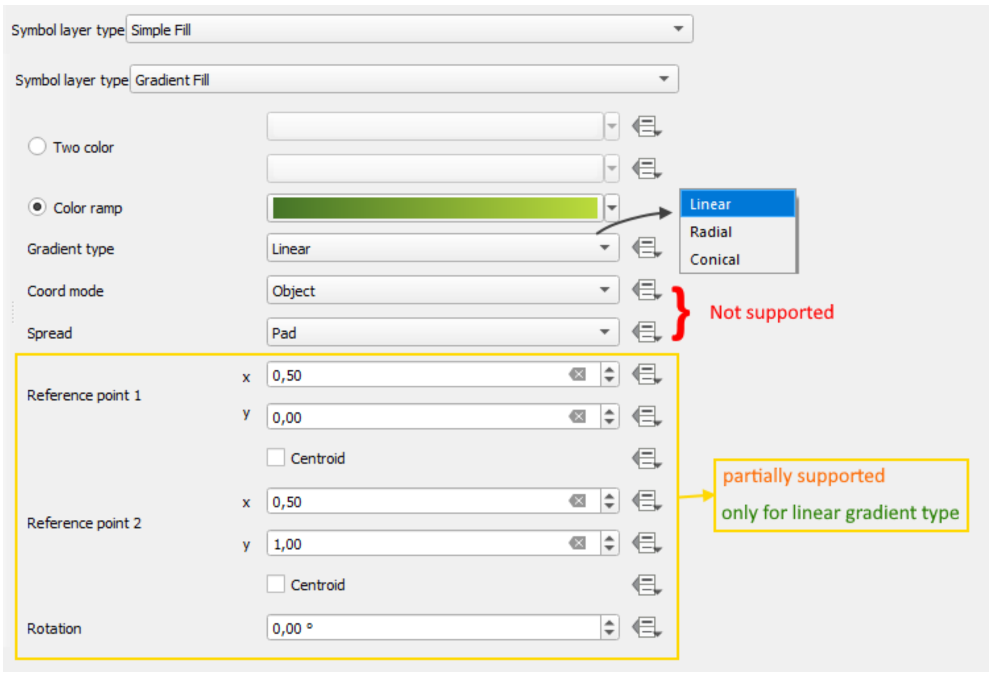
SVG Groups
- accommodation
- amenity
- arrows
- backgrounds
- components
- crosses
- emergency
- entertainment
- food
- gpsicons
- health
- hydro

SVG Images

Figure 37 – SVG template fill

5.2.6.4. Gradient fill

By publishing of map project as service in **CoGIS Server**, the gradient fill is supported, see Figure 38.



Symbol layer type: Simple Fill
Symbol layer type: Gradient Fill
( ) Two color
(●) Color ramp
Gradient type: Linear
Coord mode: Object
Spread: Pad
Reference point 1: x 0,50; y 0,00; ☐ Centroid
Reference point 2: x 0,50; y 1,00; ☐ Centroid
Rotation: 0,00 °

Figure 38 – Gradient fill

5.2.6.5. Outline: marker polyline

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

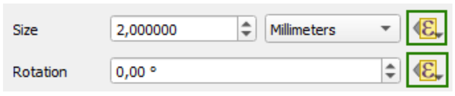
5.2.6.6. Outline: simple polyline

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

5.2.7. Redefining data (Data defined override)

CoGIS Server 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 39.



Size: 2,000000 Millimeters
Rotation: 0,00 °

Figure 39 – Setting size and rotation angle

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

CoGIS Server 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", "exp", "sqrt", "ltrim", "rtrim", "substr", "substring", "concat", "lower", "upper", "pow", "power", "andbits", "len", "length", "coalesce", "mod", "scale_exp", "scale_linear", "to_string", "tostring", "var", "min", "max", "now", "interval", "date_part"*, see Figure 40.

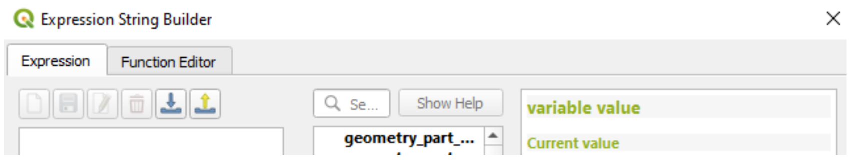


Figure 40 – SQL expressions specified at the QGIS project level

5.2.9. General symbology settings

The below figure shows general settings of symbology supported by publishing of map project as service in **CoGIS Server**, see Figure 41.

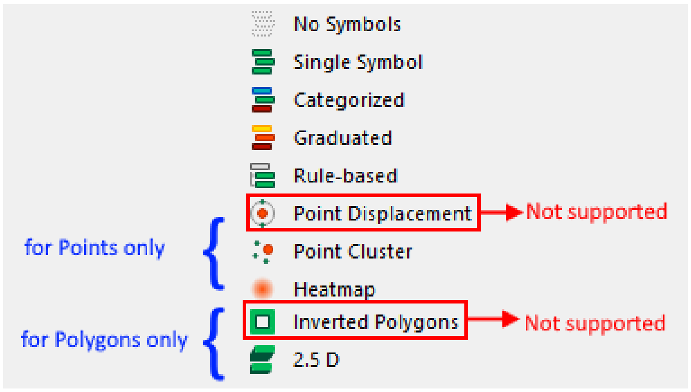


Figure 41 – General settings of symbology supported by publishing of map project as service

5.2.9.1. Symbology setting: no symbols

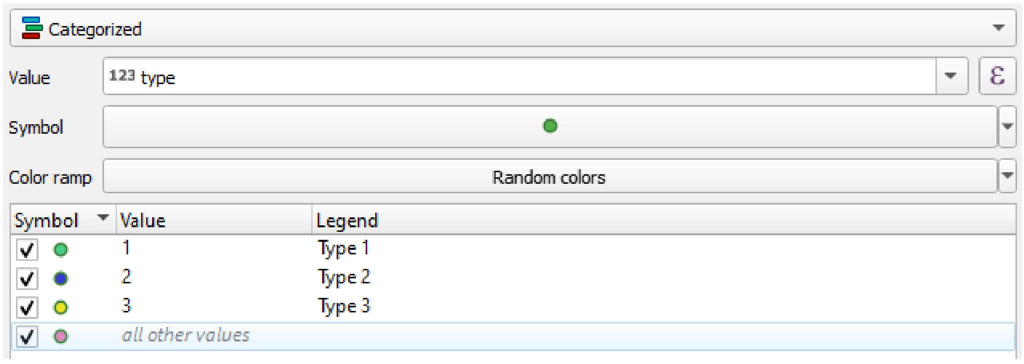
The symbols for features of this layer will not be displayed.

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 42.


Categorized

Value: 123 type

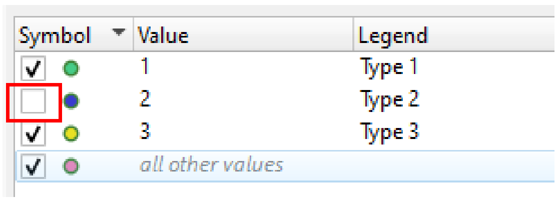
Symbol: ●

Color ramp: Random colors

Symbol	Value	Legend
✓ ●	1	Type 1
✓ ●	2	Type 2
✓ ●	3	Type 3
✓ ●	*all other values*	

Figure 42 – Symbology setting: unique values

At that, if you turn off visibility of one of the values in QGIS project, see Figure 43, then in the published service in **CoGIS Server** and **CoGIS Portal** 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.



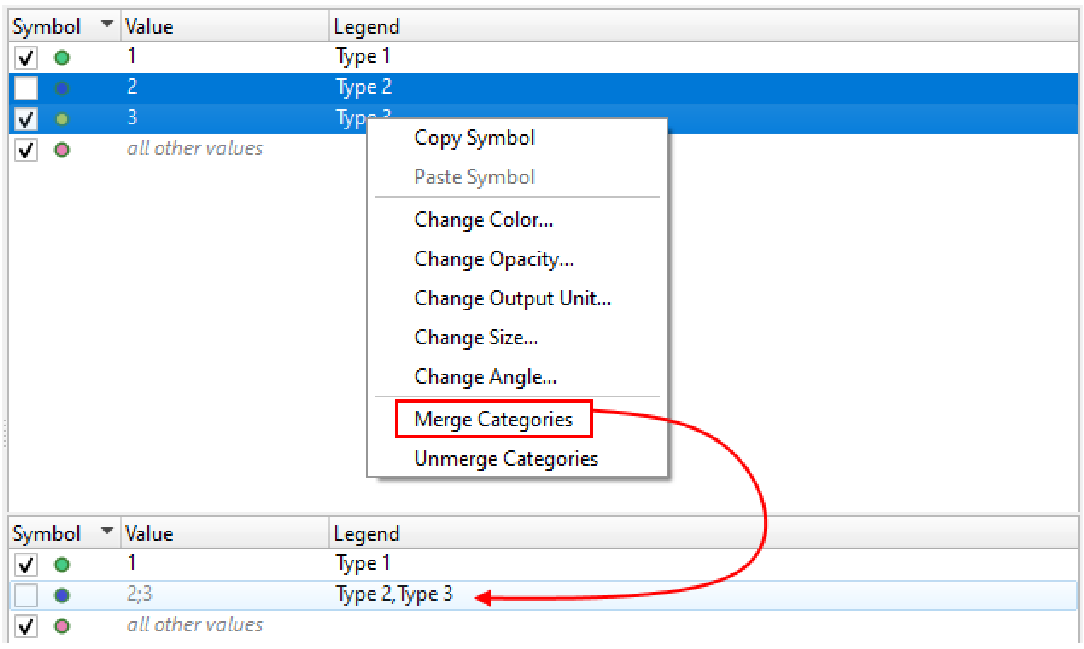
Symbol	Value	Legend
✓ ●	1	Type 1
●	2	Type 2
✓ ●	3	Type 3
✓ ●	*all other values*	

Figure 43 – 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 44.



Symbol	Value	Legend
✓ ●	1	Type 1
●	2	Type 2
✓ ●	3	Type 3
✓ ●	*all other values*	

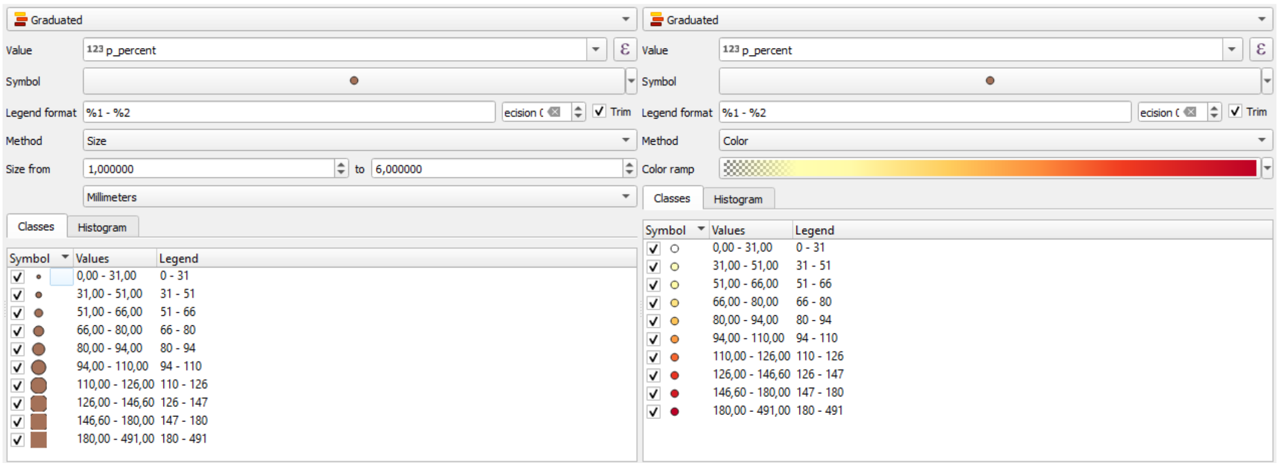
Symbol	Value	Legend
✓ ●	1	Type 1
●	2;3	Type 2, Type 3
✓ ●	*all other values*	

Figure 44 – 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 45.

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



Graduated

Value: 123 p_percent

Symbol: ●

Legend format: %1 - %2 ✓ Trim

Method: Size

Size from: 1,000000 to 6,000000

Millimeters

Classes | Histogram

Symbol	Values	Legend
✓ ●	0,00 - 31,00	0 - 31
✓ ●	31,00 - 51,00	31 - 51
✓ ●	51,00 - 66,00	51 - 66
✓ ●	66,00 - 80,00	66 - 80
✓ ●	80,00 - 94,00	80 - 94
✓ ●	94,00 - 110,00	94 - 110
✓ ●	110,00 - 126,00	110 - 126
✓ ●	126,00 - 146,60	126 - 147
✓ ●	146,60 - 180,00	147 - 180
✓ ●	180,00 - 491,00	180 - 491

Graduated

Value: 123 p_percent

Symbol: ●

Legend format: %1 - %2 ✓ Trim

Method: Color

Color ramp

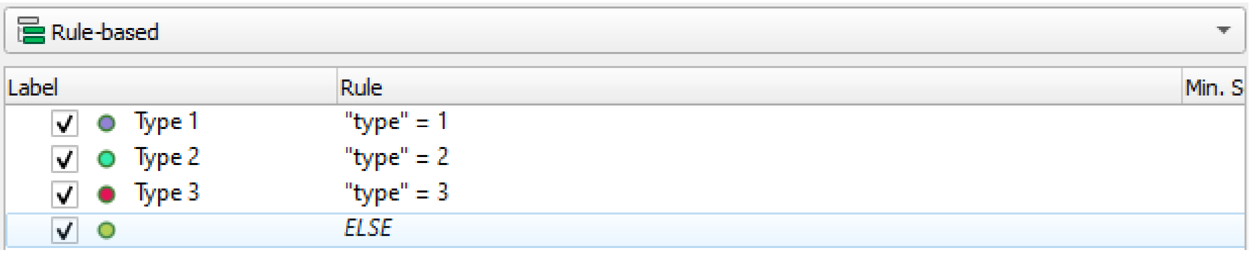
Classes | Histogram

Symbol	Values	Legend
✓ ○	0,00 - 31,00	0 - 31
✓ ●	31,00 - 51,00	31 - 51
✓ ●	51,00 - 66,00	51 - 66
✓ ●	66,00 - 80,00	66 - 80
✓ ●	80,00 - 94,00	80 - 94
✓ ●	94,00 - 110,00	94 - 110
✓ ●	110,00 - 126,00	110 - 126
✓ ●	126,00 - 146,60	126 - 147
✓ ●	146,60 - 180,00	147 - 180
✓ ●	180,00 - 491,00	180 - 491

Figure 45 – Graduated symbol

5.2.9.5. Symbol setting: rules

CoGIS Server supports symbols set for the layer in the QGIS project via the SQL filter, see Figure 46.

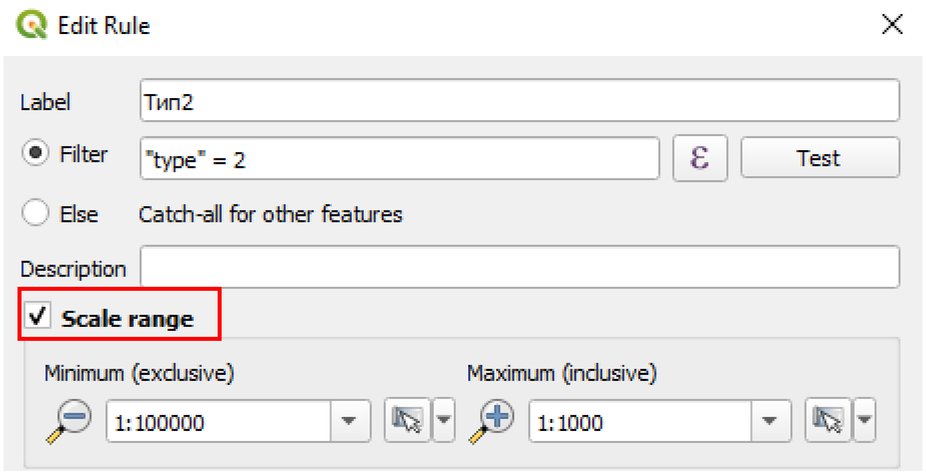


Rule-based

Label	Rule	Min. S
✓ ● Type 1	"type" = 1	
✓ ● Type 2	"type" = 2	
✓ ● Type 3	"type" = 3	
✓ ●	*ELSE*	

Figure 46 – 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 Portal** 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 47.



Edit Rule

Label: Тип2

Filter: "type" = 2 | Test

Else: Catch-all for other features

Description:

✓ **Scale range**

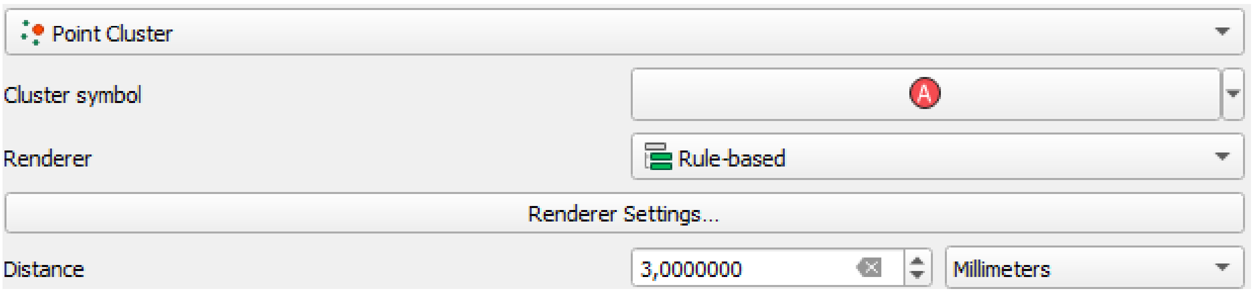
Minimum (exclusive): 1:100000

Maximum (inclusive): 1:1000

Figure 47 – Visibility within the scale

5.2.9.6. Symbol setting: clusterization

CoGIS Server supports the options of features grouping and display of group in the group centroid specified at the level of QGIS project, see Figure 48.



Point Cluster

Cluster symbol	
Renderer	Rule-based
Distance	3,0000000 Millimeters

Figure 48 – 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 49.

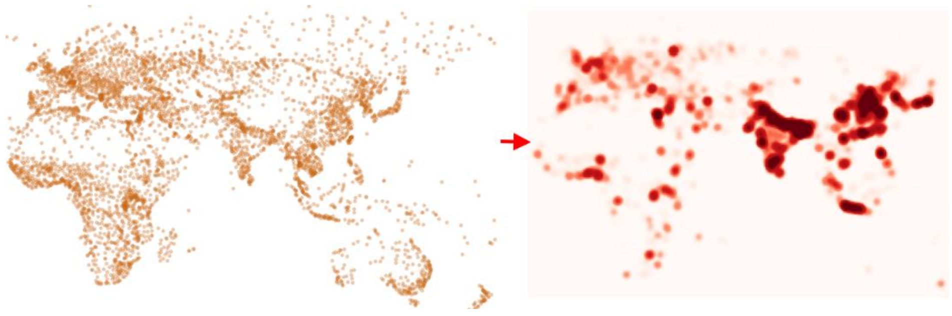
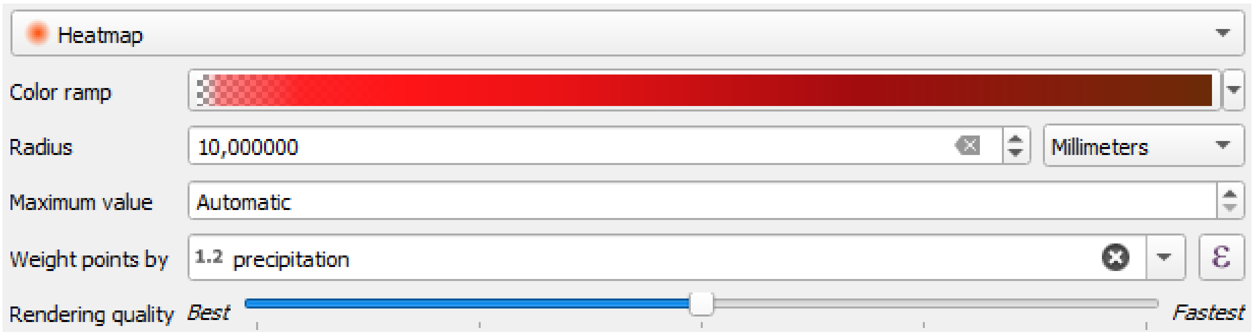


Figure 49 – Example of heat map

CoGIS Server supports layer symbology as heat maps and interpolation maps set at the level of QGIS project, see Figure 50.



Heatmap

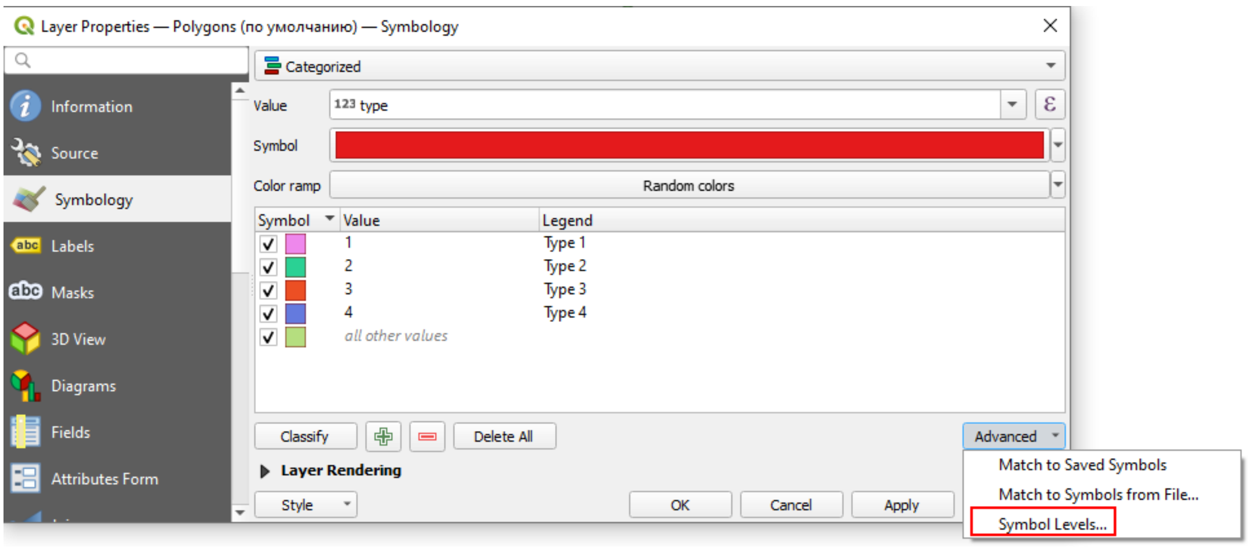
Color ramp	
Radius	10,000000 Millimeters
Maximum value	Automatic
Weight points by	1.2 precipitation
Rendering quality	Best – Fastest

Figure 50 – Setting layer symbology as heat map

For additional setting of interpolation maps display in **CoGIS Server**, you can set properties in the variables settings, see section 6.8 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 51.



Layer Properties — Polygons (по умолчанию) — Symbology

Categorized

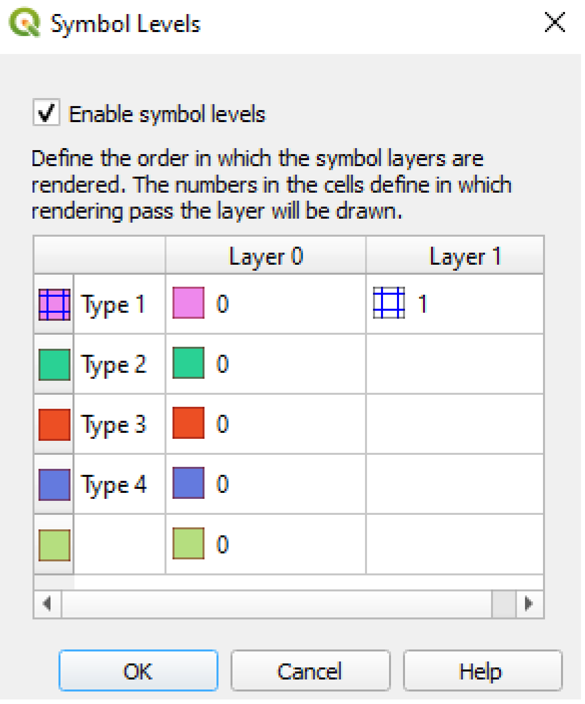
Value: 123 type

Color ramp: Random colors

Symbol	Value	Legend
✓ 🟪	1	Type 1
✓ 🟩	2	Type 2
✓ 🟥	3	Type 3
✓ 🟦	4	Type 4
✓ 🟩	*all other values*	

Figure 51 – 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 52.



Symbol Levels

✓ Enable symbol levels

Define the order in which the symbol layers are rendered. The numbers in the cells define in which rendering pass the layer will be drawn.

	Layer 0	Layer 1
▦ Type 1	🟪 0	▦ 1
🟩 Type 2	🟩 0	
🟥 Type 3	🟥 0	
🟦 Type 4	🟦 0	
🟩	🟩 0	

Figure 52 – Symbol levels (2)

5.2.10. Layer rendering

CoGIS Server supports some parameters of layer rendering set at the level of map project in QGIS, see Figure 53.

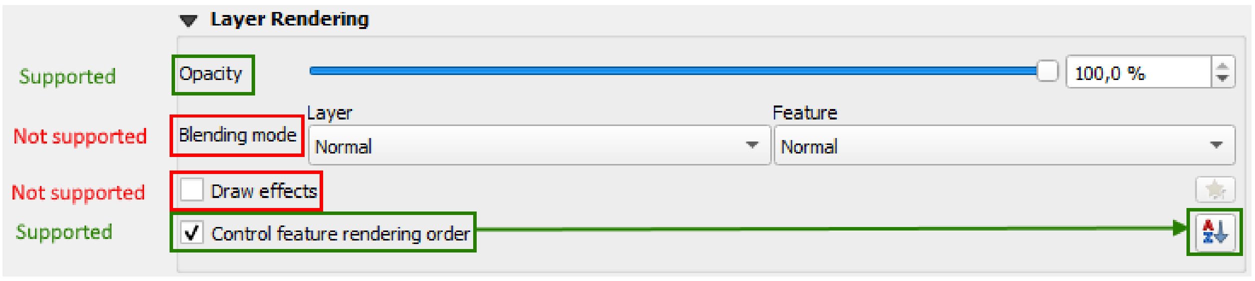
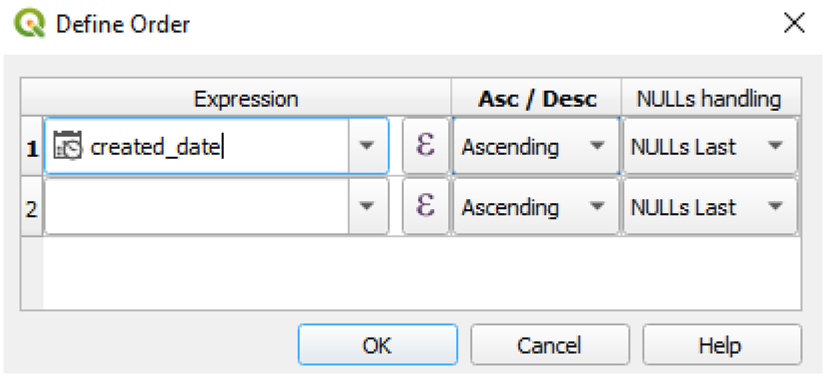


Figure 53 – Parameters of layer rendering

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

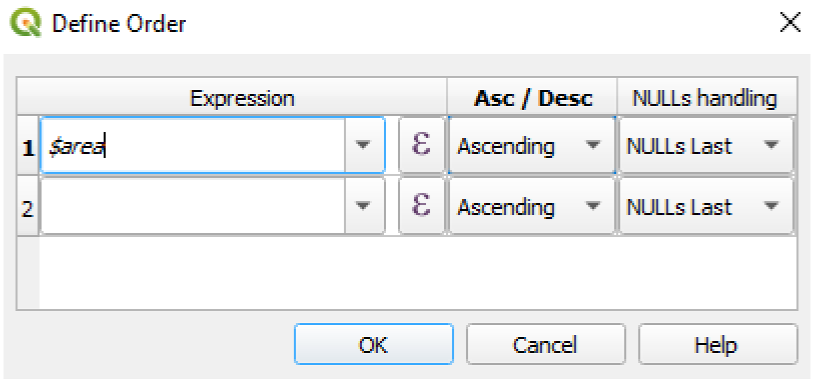
Order of features rendering – this property allows you to set sorting of the layer features by specific field or expression, see Figure 54.


Define Order

	Expression	Asc / Desc	NULLs handling
1	created_date	Ascending	NULLs Last
2		Ascending	NULLs Last

Figure 54 – 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 55.

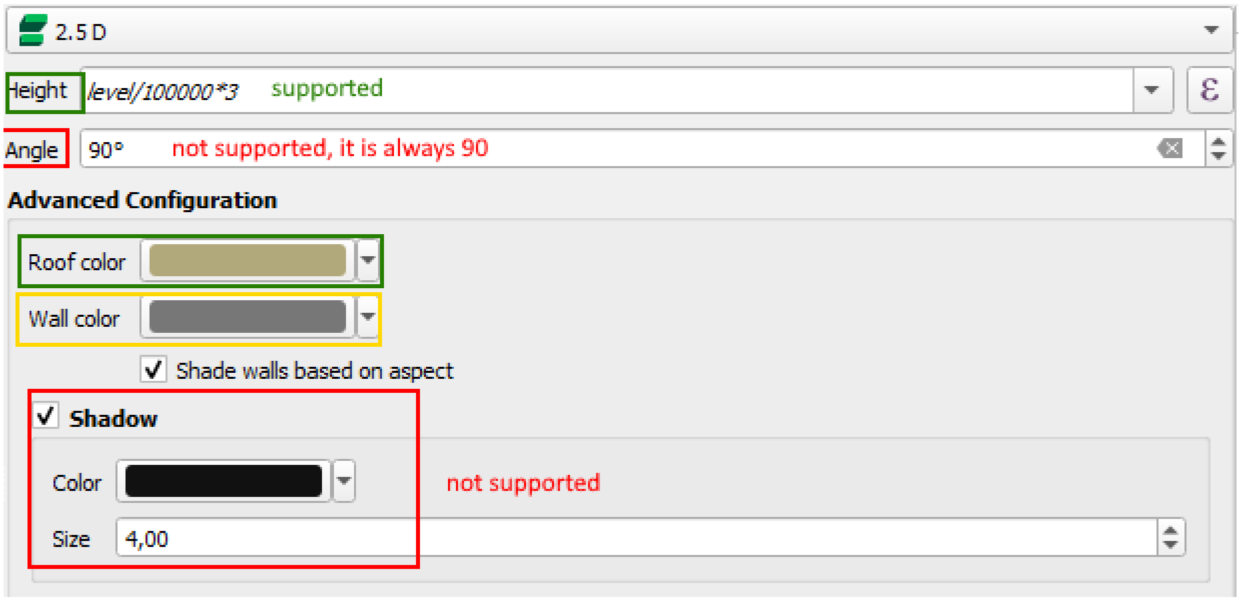

Define Order

	Expression	Asc / Desc	NULLs handling
1	$area	Ascending	NULLs Last
2		Ascending	NULLs Last

Figure 55 – Example of sorting features by area

5.2.11. 2.5D (Pseudo volume)

CoGIS Server supports layer symbology in the form of a pseudo-volume specified at the QGS project. This is the ability to display pseudo 3D features based on the areal features with specified height. The main application of this option is the display of 3D buildings based on their height (number of storeys), see Figure Figure *56*7.


2.5 D

Height *level/100000*3* supported

Angle 90° not supported, it is always 90

Advanced Configuration

Roof color

Wall color

✓ Shade walls based on aspect

✓ **Shadow**

Color

Size 4,00

not supported

Figure 56 - 2.5D Pseudo volume

The height is defined in the map units, the simple formulas are supported. If an integer is divided by an integer, then according to the standard it will always be 0, so it is recommended to divide by a fractional number (100000.0 in the example). The color of the roof is supported, only the transparency of the wall is taken from the color of the wall, and the color of the walls is taken from the color of the roof.

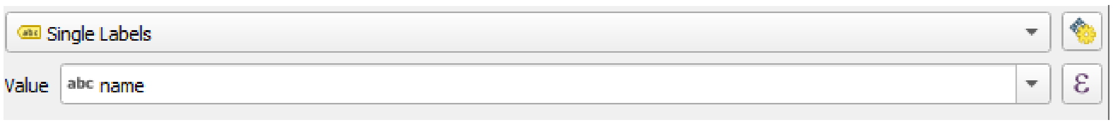
Note: this standard QGIS setting is a simplified analogue of setting using variables, described in section 6.10.

5.3. Labels

5.3.1. Methods of specifying labels

CoGIS Server supports the following methods of specifying labels set at the level of map project in QGIS:

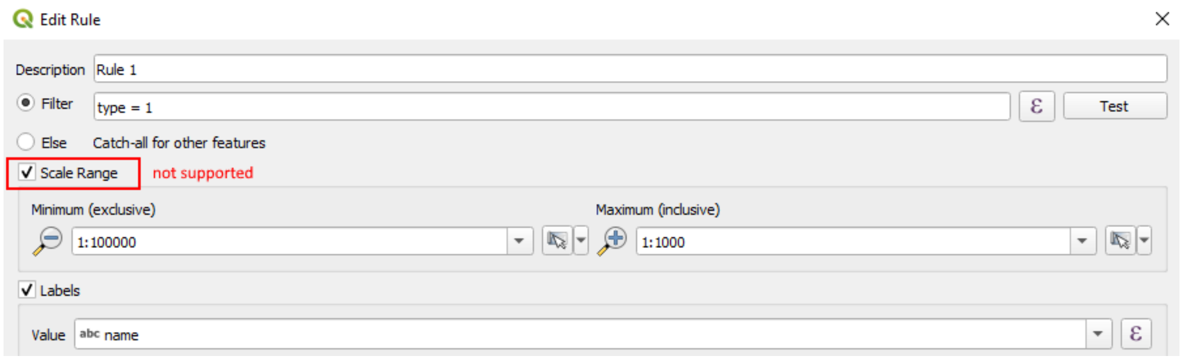
- No labels;
- Single label – labeling by values from selected field, see Figure 57;


Single Labels

Value abc name

Figure 57 – Single label

- Labels based on rules – label visibility is specified based on SQL filter and label text – based on SQL expression, see Figure 58.



Edit Rule

Description: Rule 1

Filter: type = 1

Else: Catch-all for other features

✓ Scale Range — not supported

Minimum (exclusive): 1:100000

Maximum (inclusive): 1:1000

✓ Labels

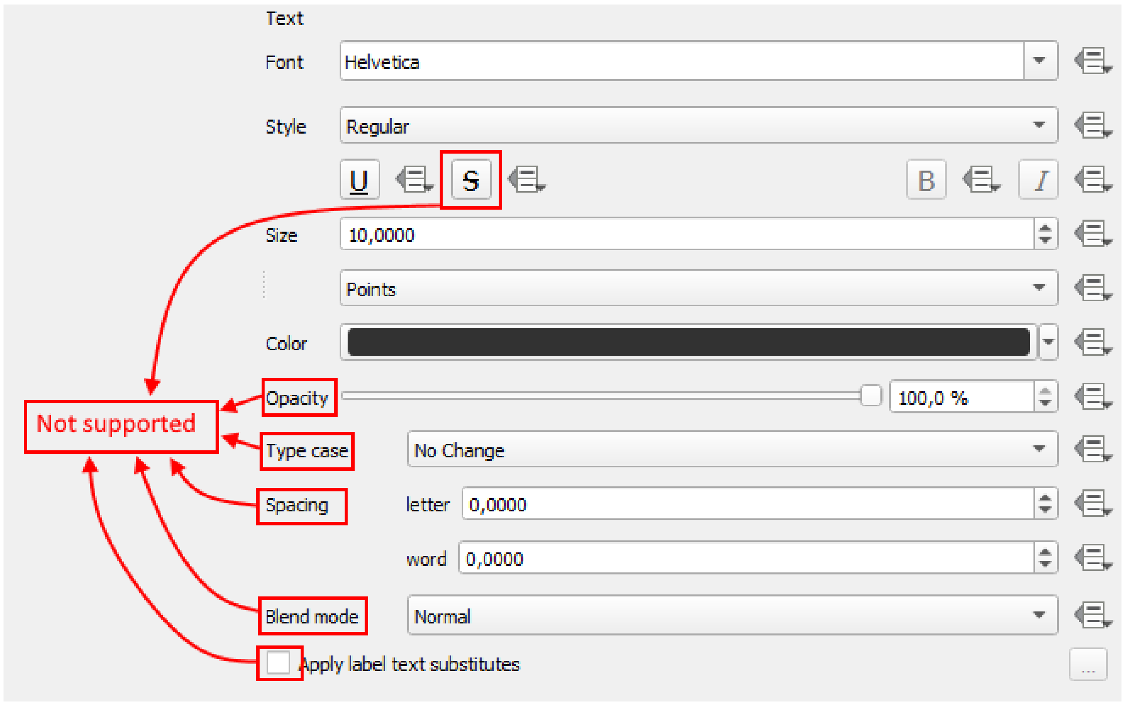
Value: abc name

Figure 58 – Labels based on rules

5.3.2. Settings

CoGIS Server supports the following labels settings specified at the level of map project in QGIS:

- Text, see Figure 59;



Text

Font: Helvetica

Style: Regular

Size: 10,0000

Points

Color

Opacity: 100,0 %

Type case: No Change

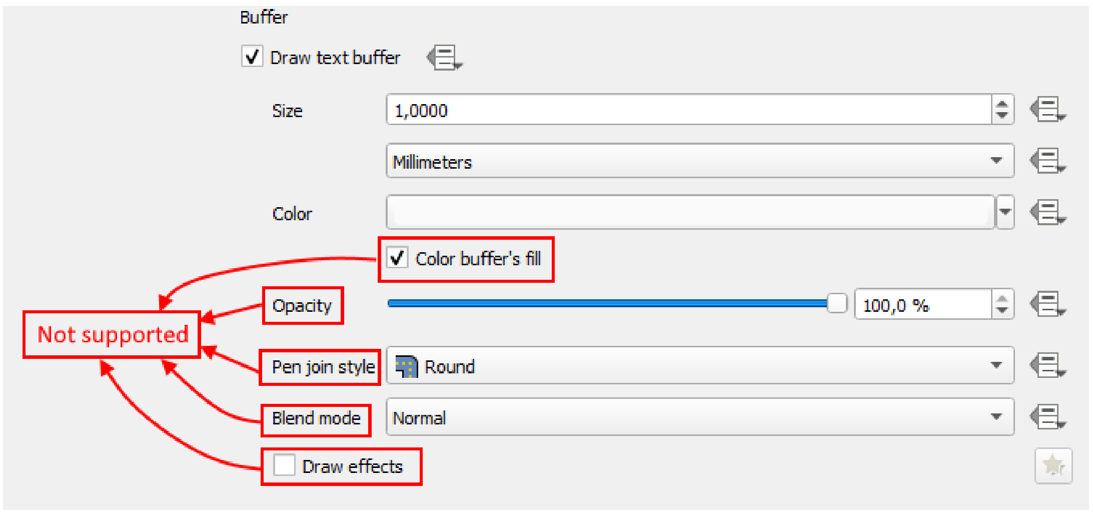
Spacing: letter 0,0000; word 0,0000

Blend mode: Normal

Apply label text substitutes

Figure 59 – Settings for label text

- Buffer, see Figure 60;



Buffer

✓ Draw text buffer

Size 1,0000

Millimeters

Color

✓ Color buffer's fill

Opacity 100,0 %

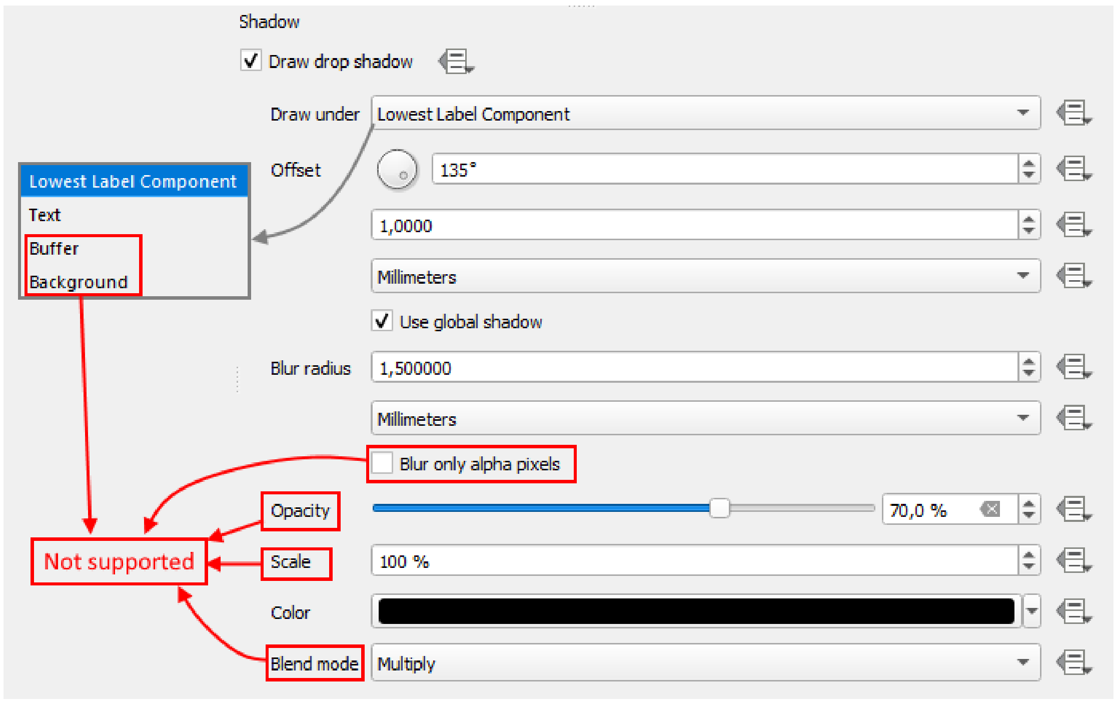
Pen join style Round

Blend mode Normal

Draw effects

Figure 60 – Settings for label buffer

- Shadow, see Figure 61;



Shadow

✓ Draw drop shadow

Draw under Lowest Label Component

Offset 135°

1,0000

Millimeters

✓ Use global shadow

Blur radius 1,500000

Millimeters

Blur only alpha pixels

Opacity 70,0 %

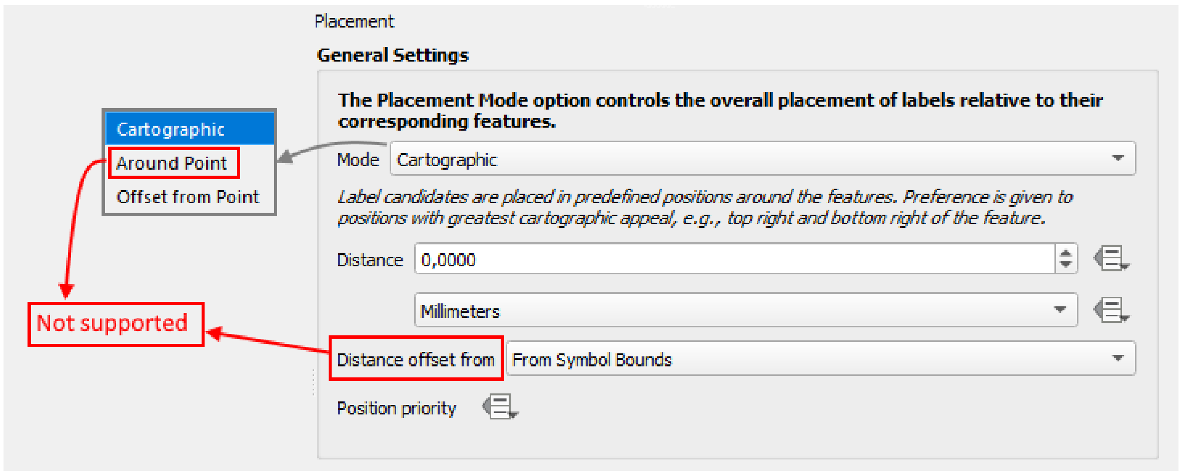
Scale 100 %

Color

Blend mode Multiply

Figure 61 – Settings for label shadow

- Location for point feature – only *Cartographic* option is supported by default, see Figure 62.


Placement

General Settings

The Placement Mode option controls the overall placement of labels relative to their corresponding features.

Mode: Cartographic

Label candidates are placed in predefined positions around the features. Preference is given to positions with greatest cartographic appeal, e.g., top right and bottom right of the feature.

Distance: 0,0000

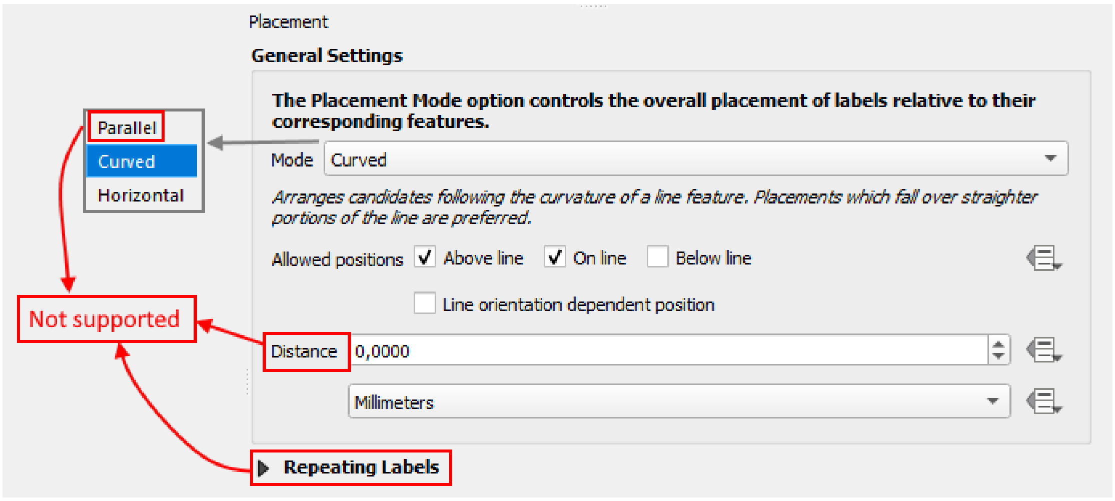
Millimeters

Distance offset from: From Symbol Bounds

Position priority

Figure 62 – 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 63;


Placement

General Settings

The Placement Mode option controls the overall placement of labels relative to their corresponding features.

Mode: Curved

Arranges candidates following the curvature of a line feature. Placements which fall over straighter portions of the line are preferred.

Allowed positions: ✓ Above line ✓ On line ☐ Below line

☐ Line orientation dependent position

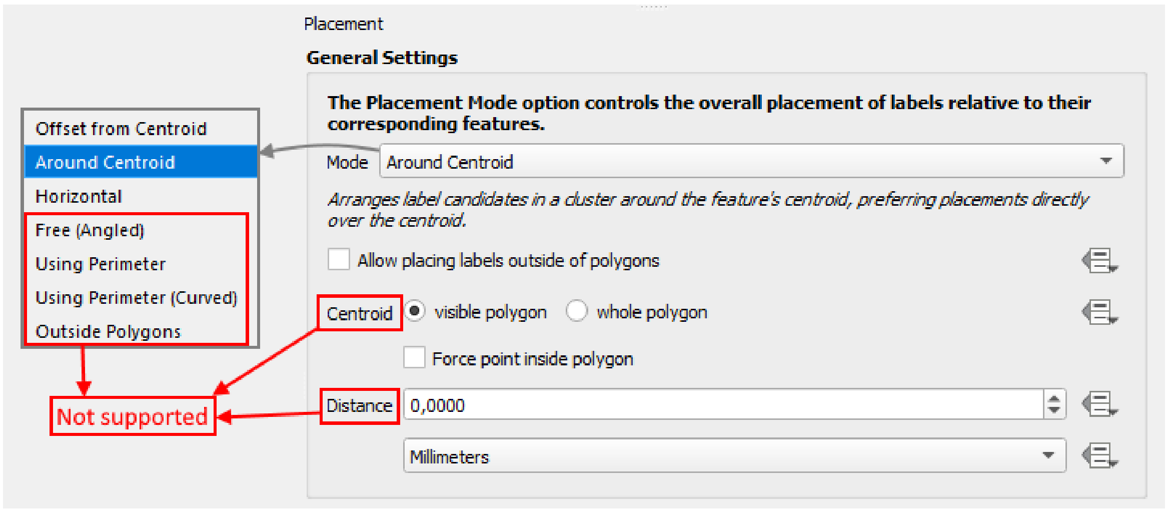
Distance: 0,0000

Millimeters

Repeating Labels

Figure 63 – Settings for location of linear feature labels

- Location for areal feature – only location in the feature centroid is supported by default, see Figure 64;


Placement

General Settings

The Placement Mode option controls the overall placement of labels relative to their corresponding features.

Mode: Around Centroid

Arranges label candidates in a cluster around the feature's centroid, preferring placements directly over the centroid.

☐ Allow placing labels outside of polygons

Centroid: ● visible polygon ○ whole polygon

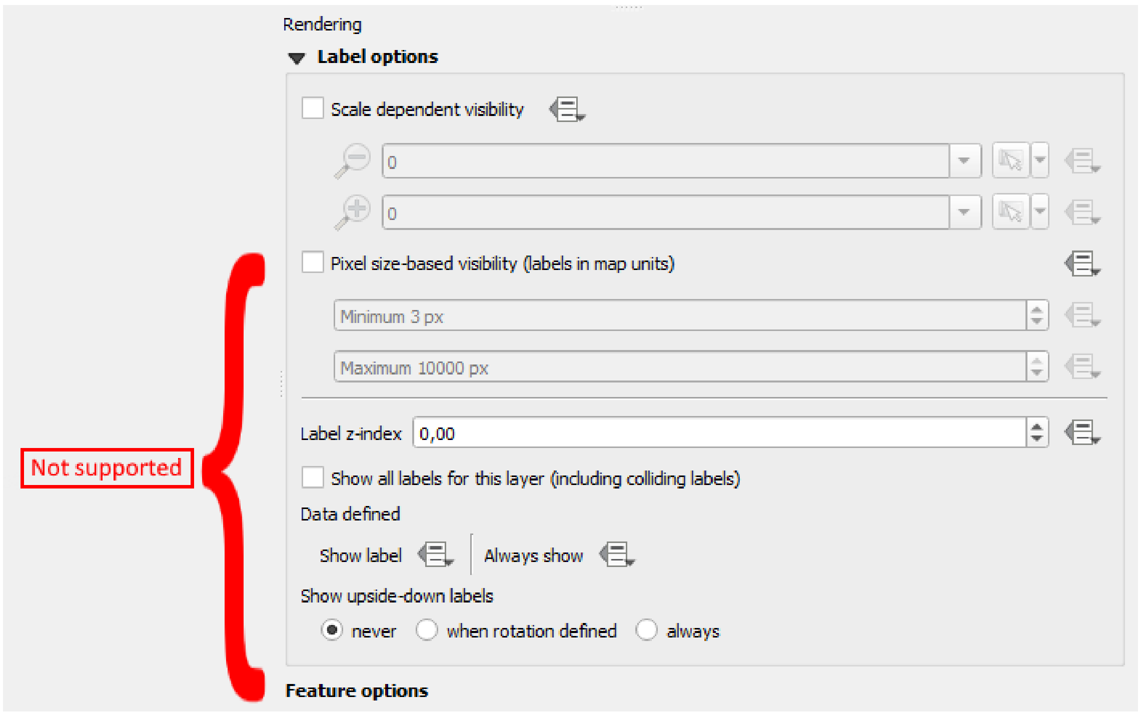
☐ Force point inside polygon

Distance: 0,0000

Millimeters

Figure 64 – Settings for location of areal feature labels

- Rendering – only visibility within scale is supported, see Figure 65;



Rendering

Label options

- [ ] Scale dependent visibility
- 0
- 0
- [ ] Pixel size-based visibility (labels in map units)
- Minimum 3 px
- Maximum 10000 px

Label z-index: 0,00

- [ ] Show all labels for this layer (including colliding labels)

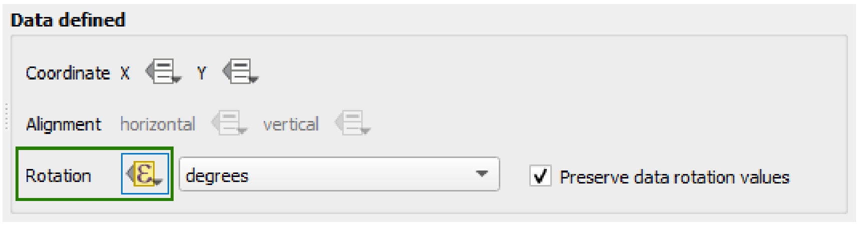
Data defined: Show label, Always show

Show upside-down labels: [✓ never] when rotation defined always

Feature options

Figure 65 – Settings for label rendering

- Rotation – supported for annotations only, see section 6.6 and Figure 66.



Data defined

Coordinate X Y

Alignment horizontal vertical

Rotation: degrees [✓] Preserve data rotation values

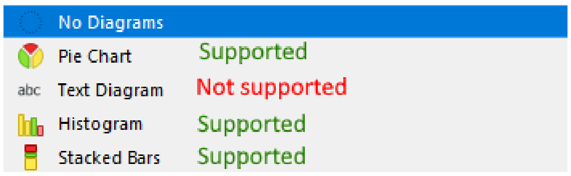
Figure 66 – Settings for rotation

Formatting and history settings are not supported.

5.4. Diagrams

CoGIS Server supports the following types of diagrams set in the QGIS project, see Figure 67:

- Pie Chart
- Histogram
- Stacked Bars

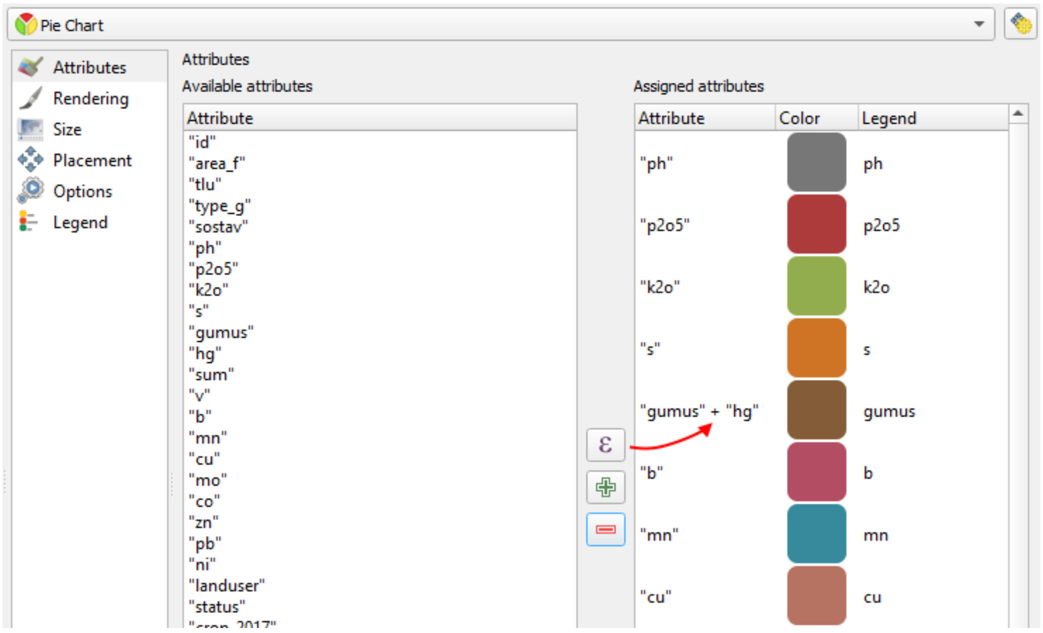


No Diagrams	
Pie Chart	Supported
Text Diagram	Not supported
Histogram	Supported
Stacked Bars	Supported

Figure 67 – Supported types of diagrams

5.4.1. Pie Chart

- Attributes: selection of pie chart slices and their colors. It is also possible to set expression, see Figure 68.



Pie Chart

Attributes

Available attributes

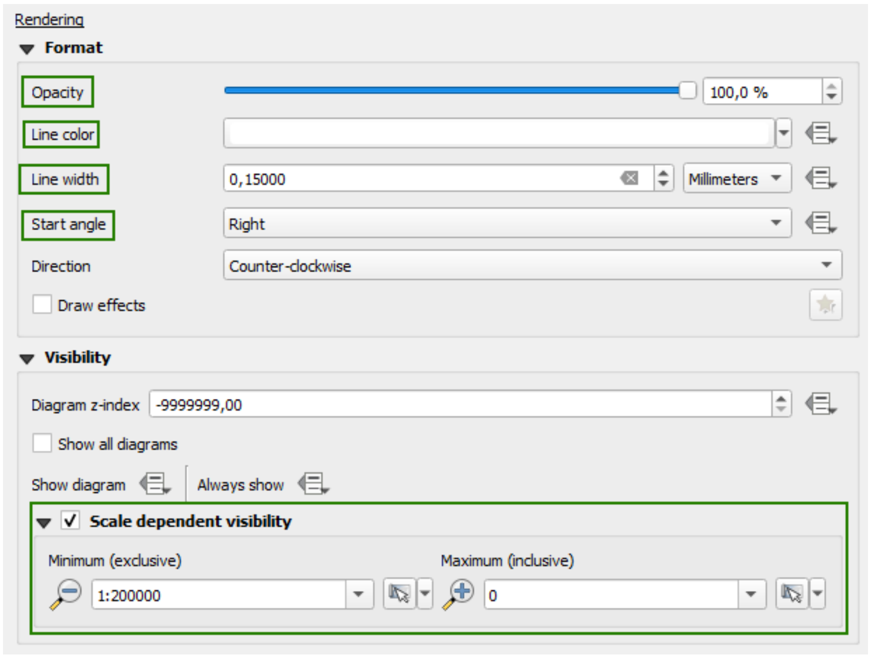
Attribute
"id"
"area_f"
"tlu"
"type_g"
"sostav"
"ph"
"p2o5"
"k2o"
"s"
"gumus"
"hg"
"sum"
"v"
"b"
"mn"
"cu"
"mo"
"co"
"zn"
"pb"
"ni"
"landuser"
"status"

Assigned attributes

Attribute	Color	Legend
"ph"	■	ph
"p2o5"	■	p2o5
"k2o"	■	k2o
"s"	■	s
"gumus" + "hg"	■	gumus
"b"	■	b
"mn"	■	mn
"cu"	■	cu

Figure 68 – Setting the pie chart attributes

- Rendering: setting transparency, pie chart slice outline, line width, chart start angle, pie chart direction (clockwise/counterclockwise), and pie chart visibility within the scale, see Figure 69.



Rendering

Format

Opacity	100,0 %
Line color	
Line width	0,15000 Millimeters
Start angle	Right
Direction	Counter-clockwise
☐ Draw effects	

Visibility

Diagram z-index -9999999,00

☐ Show all diagrams

Show diagram Always show

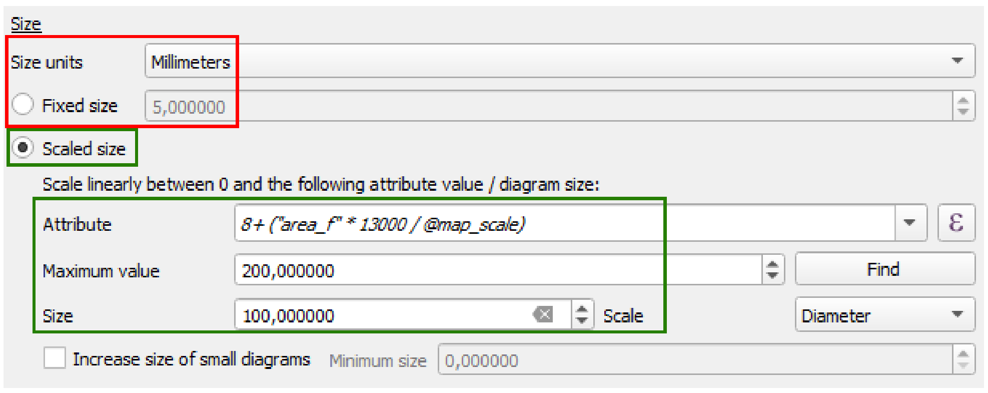
☑ **Scale dependent visibility**

Minimum (exclusive) 1:200000

Maximum (inclusive) 0

Figure 69 – Setting the pie chart rendering parameters

- Size: the Fixed size setting is not supported, but you can specify the fixed size in the Attribute field of the Scalable size option. Also in the Attribute field you can specify the expression by which the pie chart size should be calculated, the supported variable is *@map_scale* – the current map scale, see Figure 70.



Size

Size units: Millimeters

Fixed size: 5,000000

(●) Scaled size

Scale linearly between 0 and the following attribute value / diagram size:

Attribute: 8+ ("area_f" * 13000 / @map_scale)

Maximum value: 200,000000 [Find]

Size: 100,000000 Scale — Diameter

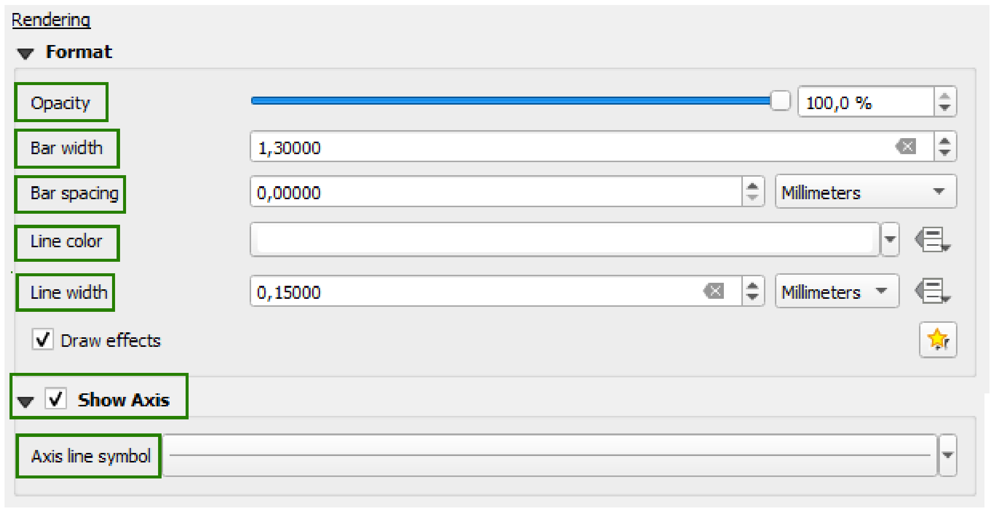
☐ Increase size of small diagrams — Minimum size: 0,000000

Figure 70 – Setting the pie chart size

- Location: the setting is not supported; the pie chart is always created by centroid of the source feature.
- The Parameter and Legend options are not used for pie charts.

5.4.2. Histogram

- Attributes: same as for Pie Chart (see above).
- Rendering: setting transparency, bar width, bar spacing, outline color and outline width, style, axis visibility and symbol, histogram visibility within the scale, see Figure 71.



Rendering

Format

Opacity: 100,0 %

Bar width: 1,30000

Bar spacing: 0,00000 Millimeters

Line color

Line width: 0,15000 Millimeters

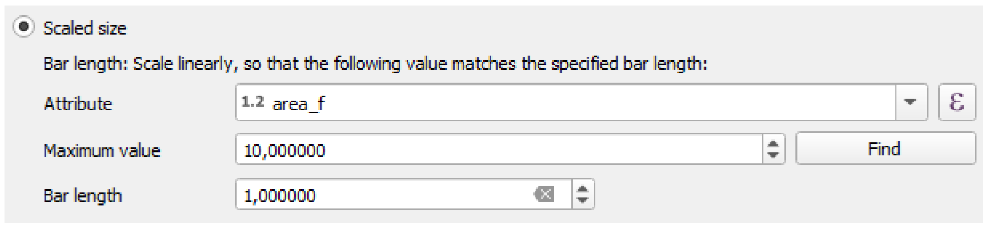
☑ Draw effects

☑ Show Axis

Axis line symbol

Figure 71 – Setting the histogram rendering parameters

- Size: setting the histogram height; only Scalable size is available. The Visibility within the scale option is not available for histograms. The Attribute field is required only for automatic calculation of maximum value, see Figure 72.



(●) Scaled size

Bar length: Scale linearly, so that the following value matches the specified bar length:

Attribute: 1.2 area_f

Maximum value: 10,000000 [Find]

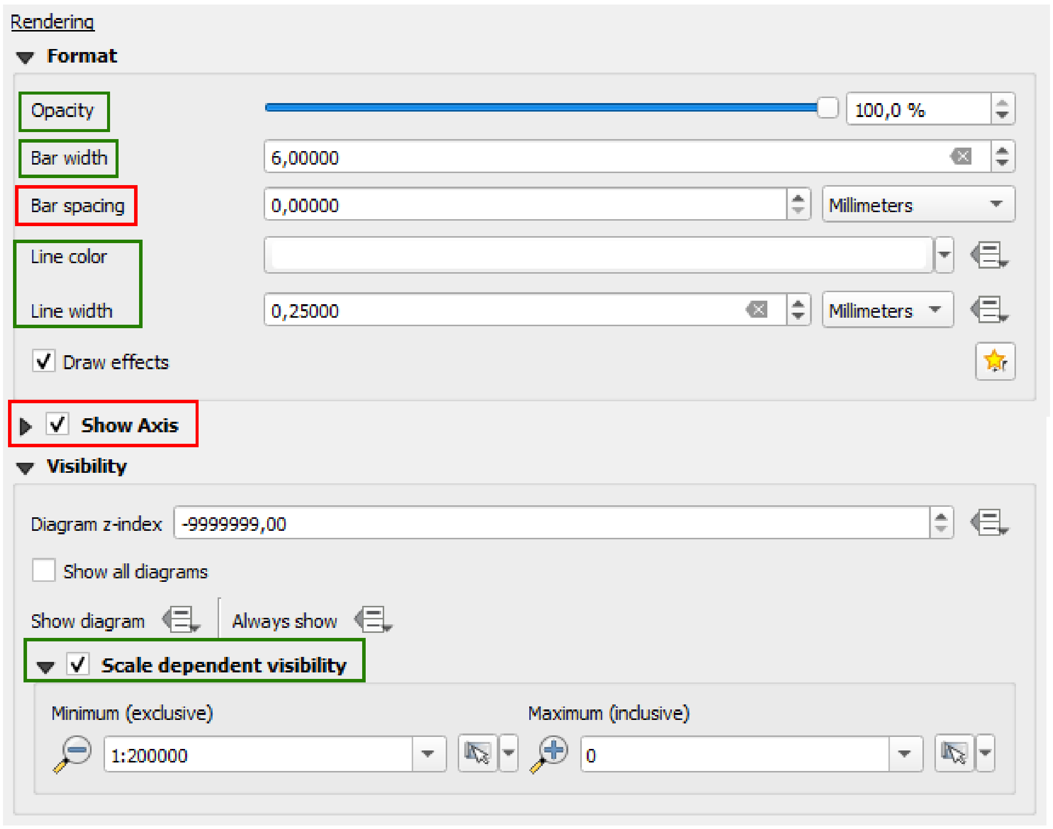
Bar length: 1,000000

Figure 72 – Setting the histogram size

- Location: this setting is not supported; the histogram is always created by centroid of the source feature.
- The Parameter and Legend options are not used for histograms.

5.4.3. Stacked Bars diagram

- Attributes: same as for Pie Chart (see above).
- Rendering: available settings are transparency, bars width, outline color and width, visibility within the scale. Not available settings are Bar spacing (bars are always close), Axis display (always without axis), see Figure 73.



Rendering

Format

Opacity 100,0 %

Bar width 6,00000

Bar spacing 0,00000 Millimeters

Line color

Line width 0,25000 Millimeters

✓ Draw effects

✓ Show Axis

Visibility

Diagram z-index -9999999,00

Show all diagrams

Show diagram Always show

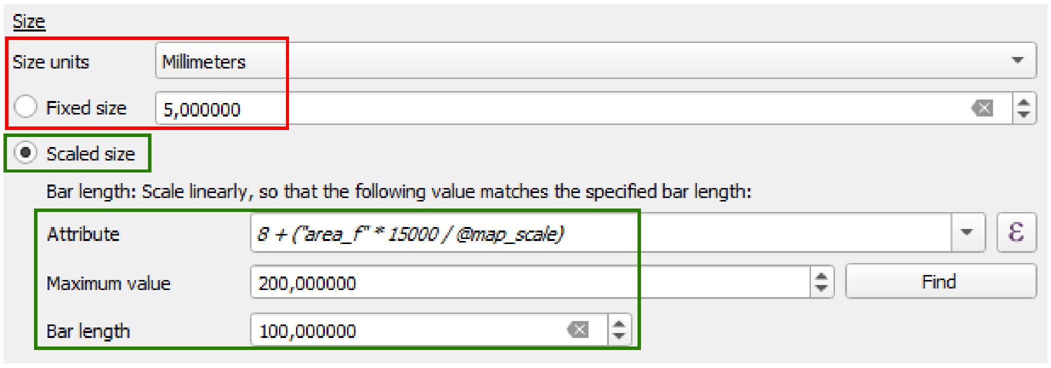
✓ Scale dependent visibility

Minimum (exclusive) 1:200000

Maximum (inclusive) 0

Figure 73 – Setting the Stacked Bars diagram rendering

- Size: the Fixed size setting is not supported, but you can specify the fixed size in the Attribute field of the Scalable size option. Also in the Attribute field you can specify the expression by which the Stacked Bars diagram size should be calculated, the supported variable is *@map_scale* – the current map scale, see Figure 74.



Size

Size units Millimeters

Fixed size 5,000000

● Scaled size

Bar length: Scale linearly, so that the following value matches the specified bar length:

Attribute *8 + ("area_f" * 15000 / @map_scale)*

Maximum value 200,000000

Bar length 100,000000

Figure 74 - Setting the Stacked Bars diagram

- Location: this setting is not supported; the Stacked Bars diagram is always created by centroid of the source feature.
- The Parameter and Legend options are not used for Stacked Bars diagrams.

5.5.Properties of layer fields (Attributes Form)

CoGIS Server supports different properties of the layer fields specified at the level of map project, see Figure 75.

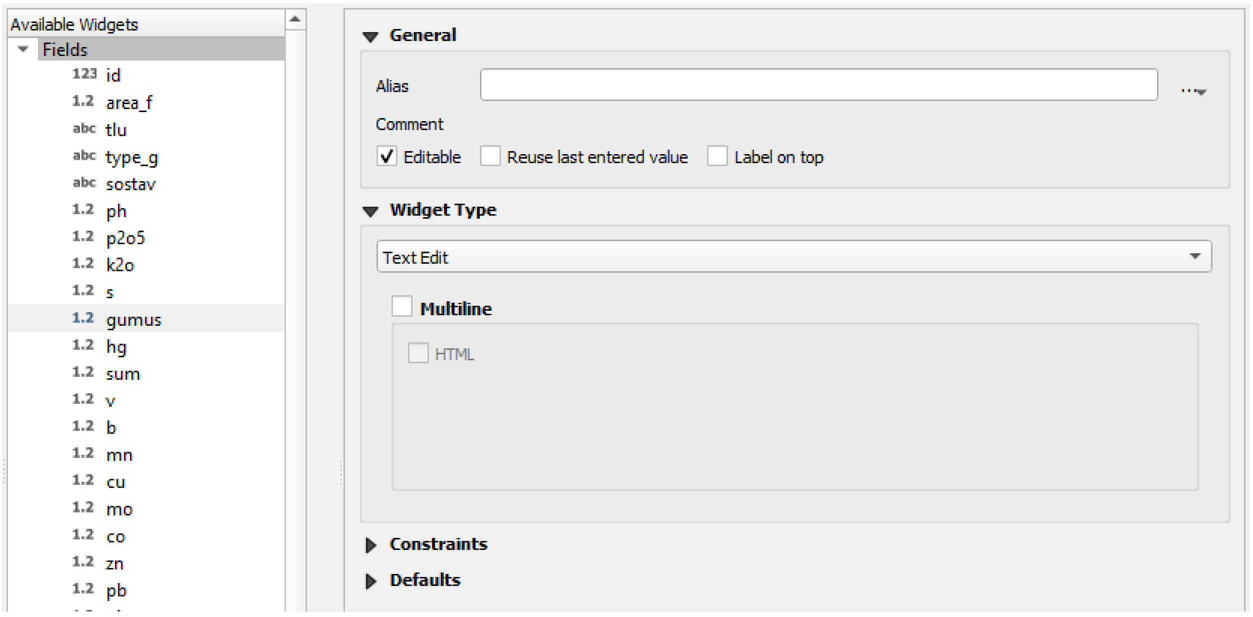


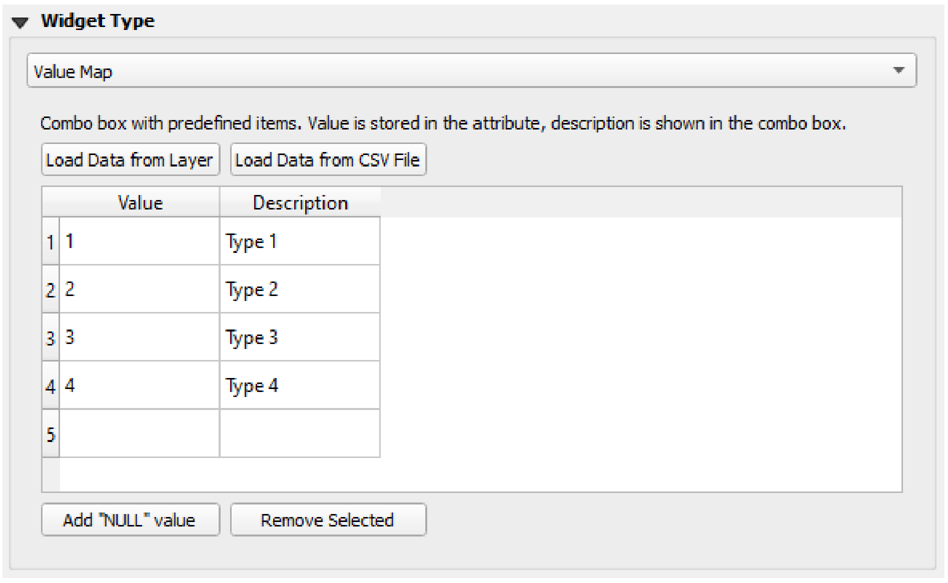
Figure 75 – Setting properties of layer fields

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 76.

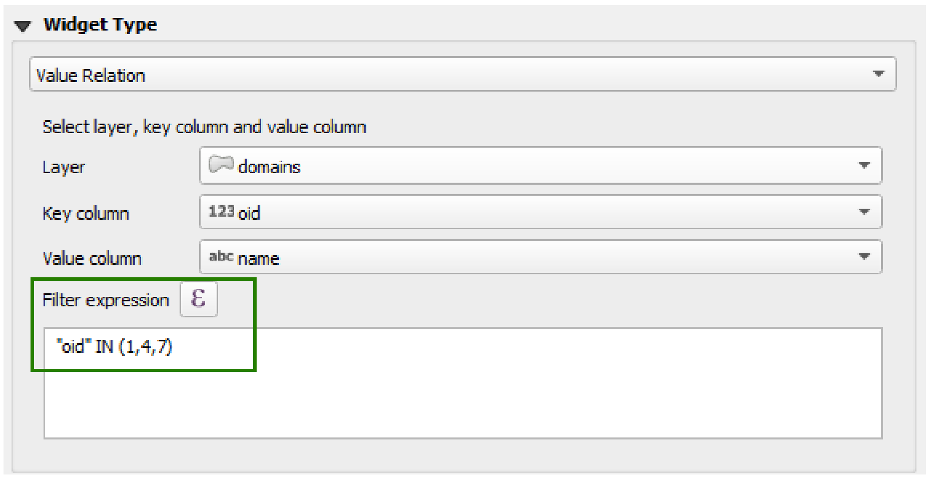


Combo box with predefined items. Value is stored in the attribute, description is shown in the combo box.

	Value	Description
1	1	Type 1
2	2	Type 2
3	3	Type 3
4	4	Type 4
5		

Figure 76 – Setting domain for the field

- ○ Related value – domain values based on values from another layer, with the filtration option, see Figure 77.


Widget Type

Value Relation

Select layer, key column and value column

Layer: domains

Key column: 123 oid

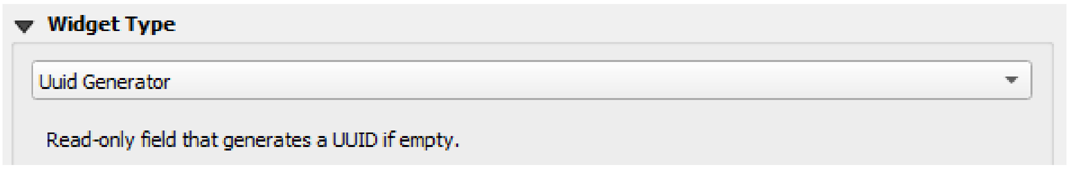
Value column: abc name

Filter expression

"oid" IN (1,4,7)

Figure 77 – Setting related values for the field

- ○ UUID generator - generation of Universally unique identifier by feature's creation, see Figure 78. The required field type is *uuid* or *varchar*.

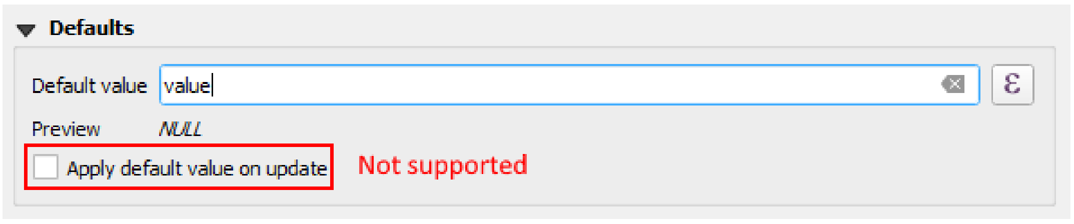

Widget Type

Uuid Generator

Read-only field that generates a UUID if empty.

Figure 78 – Setting UUID generator

- *Default values (Default expression)* – values that will be recorded to the appropriate fields by creation of the feature, see Figure 79.


Defaults

Default value: value

Preview: NULL

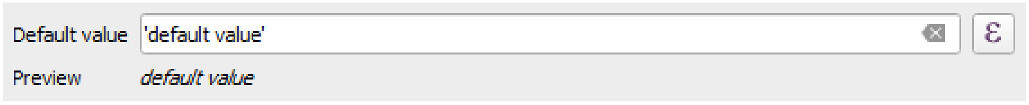
Apply default value on update

Not supported

Figure 79 – Setting default values for the field

Types of values:

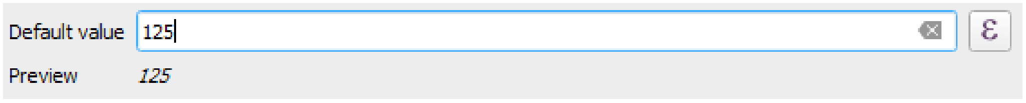
- ○ String. Field type – string, record of specified string, see Figure 80.


Default value: 'default value'

Preview: default value

Figure 80 – Default value for the string

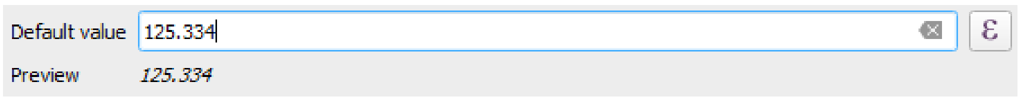
- ○ Integer. Field type – integer, record of specified integer, see Figure 81.


Default value: 125

Preview: 125

Figure 81 – Default value for the integer

- ○ Real number. Field type – real, record of specified real number, see Figure 82.


Default value: 125.334

Preview: 125.334

Figure 82 – Default value for the real number

- @user_account_name – record of the current user name in **CoGIS Server** {CurrentUser}. If the user is not authorized, *null* will be recorded. Field type - string, see Figure 83.

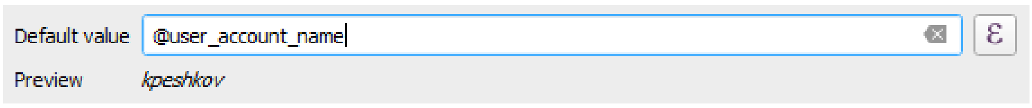

Default value @user_account_name
Preview *kpeshkov*

Figure 83 – Current user name used as the default value

- uuid() – record of the *Universally unique identifier*. Field type is *uuid* or string, see Figure 84.

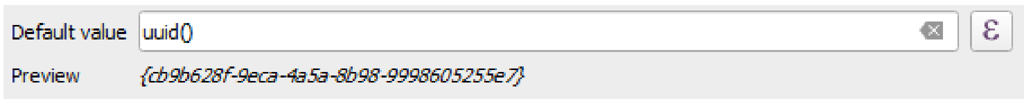

Default value uuid()
Preview *{cb9b628f-9eca-4a5a-8b98-9998605255e7}*

Figure 84 – Using the generated UUID as the default value

- now() – record of the current date-time UTC, {CurrentDateTime}. Field type - *timestamptz*, see Figure 85.

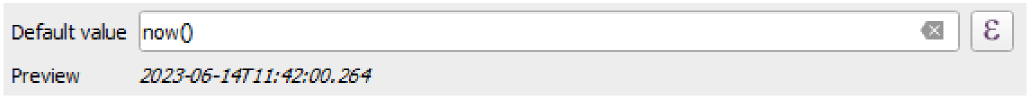

Default value now()
Preview *2023-06-14T11:42:00.264*

Figure 85 – 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 86.

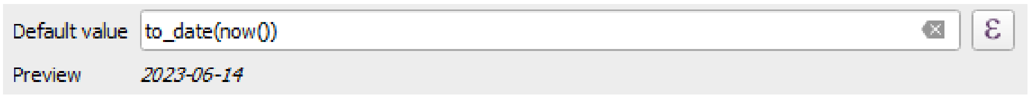

Default value to_date(now())
Preview *2023-06-14*

Figure 86 – Using the current date and time in local time zone as the default value

5.6. Joins

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

5.7. Displaying value by feature's identification

CoGIS Server 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 Portal**, see Figure 87.

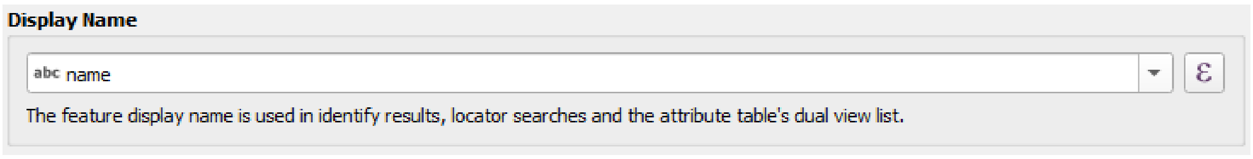

Display Name
abc name
The feature display name is used in identify results, locator searches and the attribute table's dual view list.

Figure 87 – Field values by feature's identification

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

5.8. Layer visibility settings

CoGIS Server supports settings of layer visibility in the specific scale range or in all scales, set in the map project, see Figure 88.

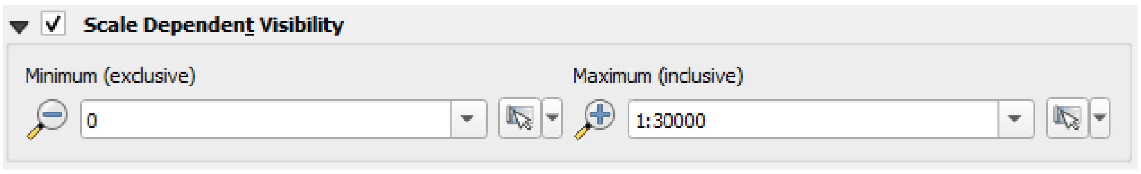


Figure 88 – Visibility settings via layer parameters

Setting/clearing the layer visibility scale ranges can be done in the context menu of the layer, see Figure 89.

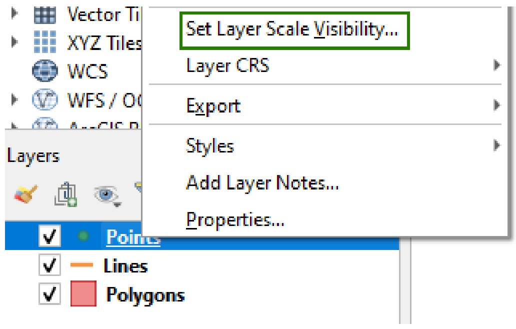
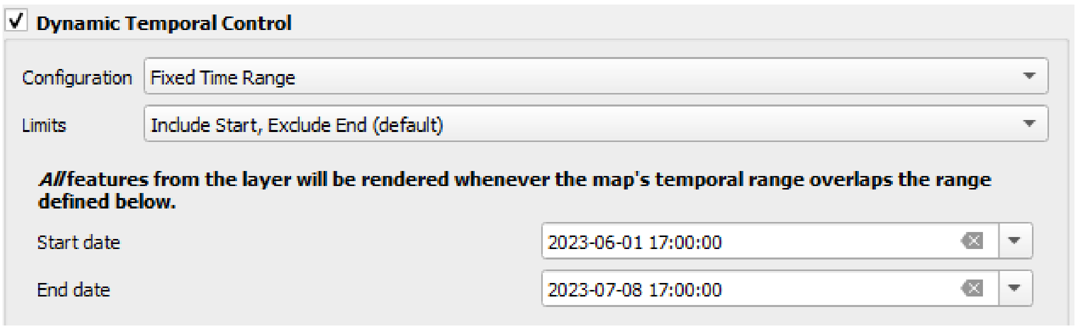


Figure 89 – Setting the layer visibility via context menu

5.9. Time-series data

CoGIS Server supports time-series data for layer. The time-series data can be configured using the following options:

- Fixed Time Range, see Figure 90.
  Start date – start date of time range
  End date – end date of time range



Dynamic Temporal Control

Configuration: Fixed Time Range

Limits: Include Start, Exclude End (default)

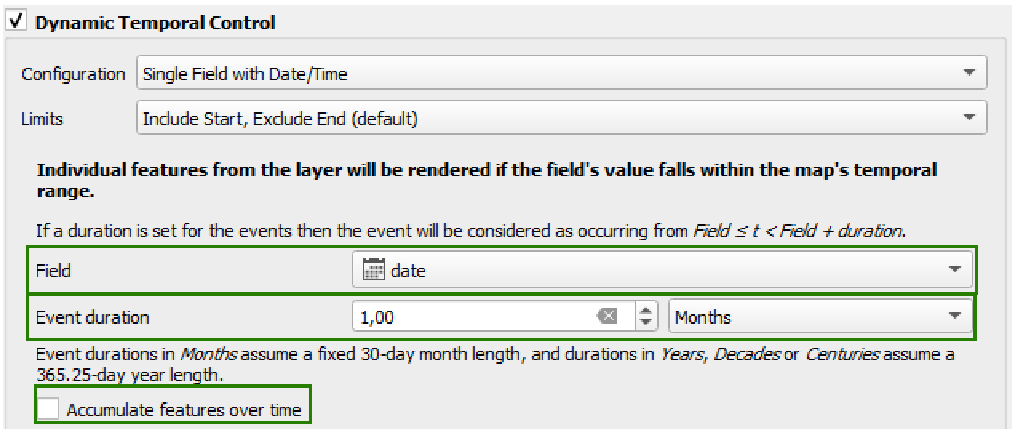
***All* features from the layer will be rendered whenever the map's temporal range overlaps the range defined below.**

Start date: 2023-06-01 17:00:00

End date: 2023-07-08 17:00:00

Figure 90 – Setting the fixed time range

- Single Field with Date/Time, see Figure 91.
  Field – selection of field from the list of fields for layer
  Event duration - the value of the time interval and the unit of this interval
  Accumulate features over time – the option enables the accumulation effect



☑ **Dynamic Temporal Control**

Configuration: Single Field with Date/Time

Limits: Include Start, Exclude End (default)

Individual features from the layer will be rendered if the field's value falls within the map's temporal range.

If a duration is set for the events then the event will be considered as occurring from *Field ≤ t < Field + duration*.

Field: date

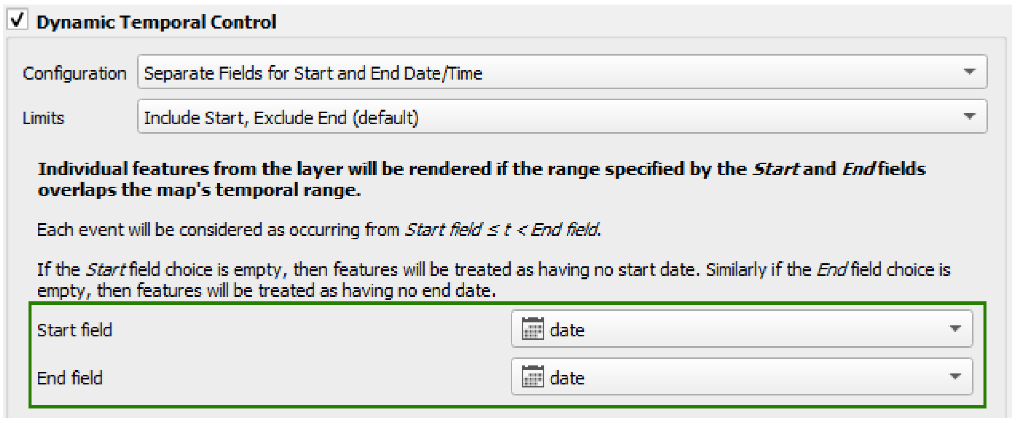
Event duration: 1,00 Months

Event durations in *Months* assume a fixed 30-day month length, and durations in *Years*, *Decades* or *Centuries* assume a 365.25-day year length.

☐ Accumulate features over time

Figure 91 – Setting the Single Field with Date/Time option

- Separate Fields for Start and End Date/Time, see Figure 92.
  Start field – the name of the field with the start date of the time range
  End field - the name of the field with the end date of the time range



☑ **Dynamic Temporal Control**

Configuration: Separate Fields for Start and End Date/Time

Limits: Include Start, Exclude End (default)

Individual features from the layer will be rendered if the range specified by the *Start* and *End* fields overlaps the map's temporal range.

Each event will be considered as occurring from *Start field ≤ t < End field*.

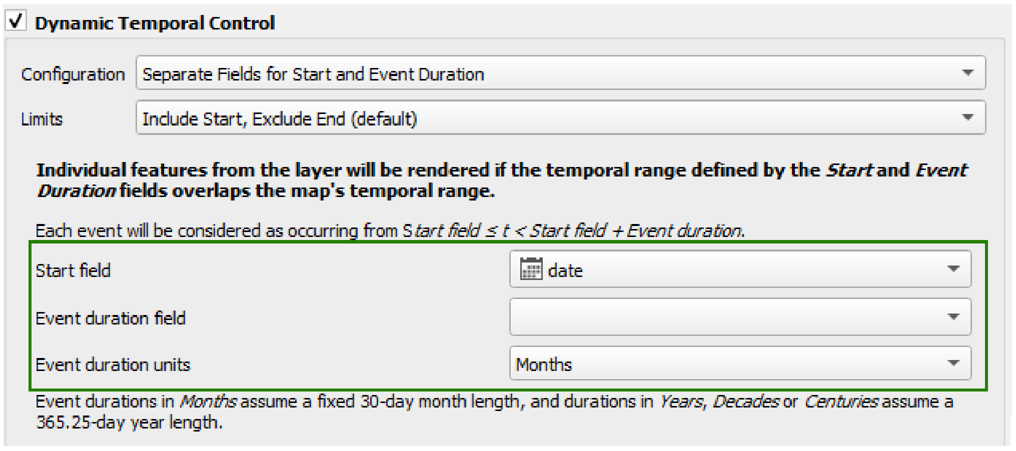
If the *Start* field choice is empty, then features will be treated as having no start date. Similarly if the *End* field choice is empty, then features will be treated as having no end date.

Start field: date

End field: date

Figure 92 – Setting the Separate Fields for Start and End Date/Time option

- Separate Fields for Start and Event Duration, see Figure 93.
  Start field - the name of the field with the start date of the time range
  Event duration field – the name of the field with the time range value
  Event duration unit – the time range units



☑ **Dynamic Temporal Control**

Configuration: Separate Fields for Start and Event Duration

Limits: Include Start, Exclude End (default)

Individual features from the layer will be rendered if the temporal range defined by the *Start* and *Event Duration* fields overlaps the map's temporal range.

Each event will be considered as occurring from *Start field ≤ t < Start field + Event duration*.

Start field: date

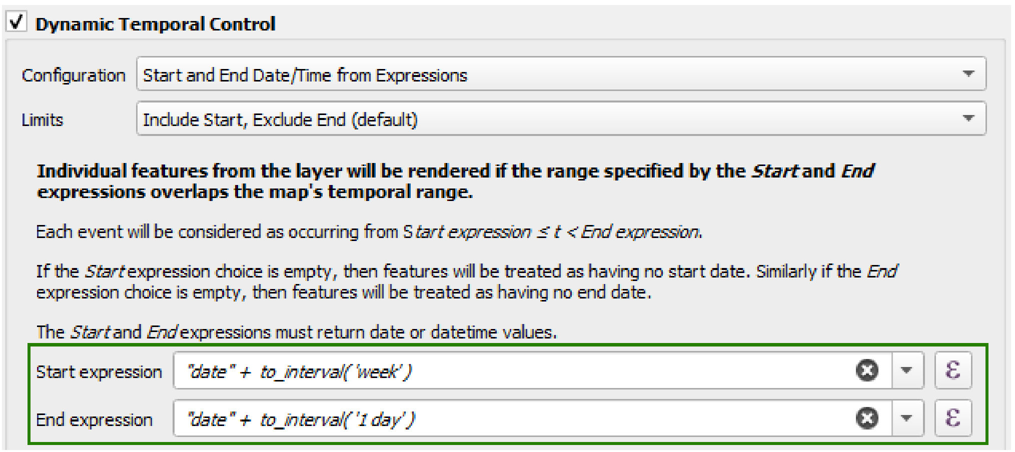
Event duration field:

Event duration units: Months

Event durations in *Months* assume a fixed 30-day month length, and durations in *Years*, *Decades* or *Centuries* assume a 365.25-day year length.

Figure 93 - Setting the Separate Fields for Start and Event Duration option

- Start and End Date/Time from Expressions, see Figure 94. The supported standard function is *to_interval* only.


☑ **Dynamic Temporal Control**

Configuration: Start and End Date/Time from Expressions

Limits: Include Start, Exclude End (default)

Individual features from the layer will be rendered if the range specified by the *Start* and *End* expressions overlaps the map's temporal range.

Each event will be considered as occurring from *Start expression ≤ t < End expression.*

If the *Start* expression choice is empty, then features will be treated as having no start date. Similarly if the *End* expression choice is empty, then features will be treated as having no end date.

The *Start* and *End* expressions must return date or datetime values.

Start expression: *"date" + to_interval('week')*

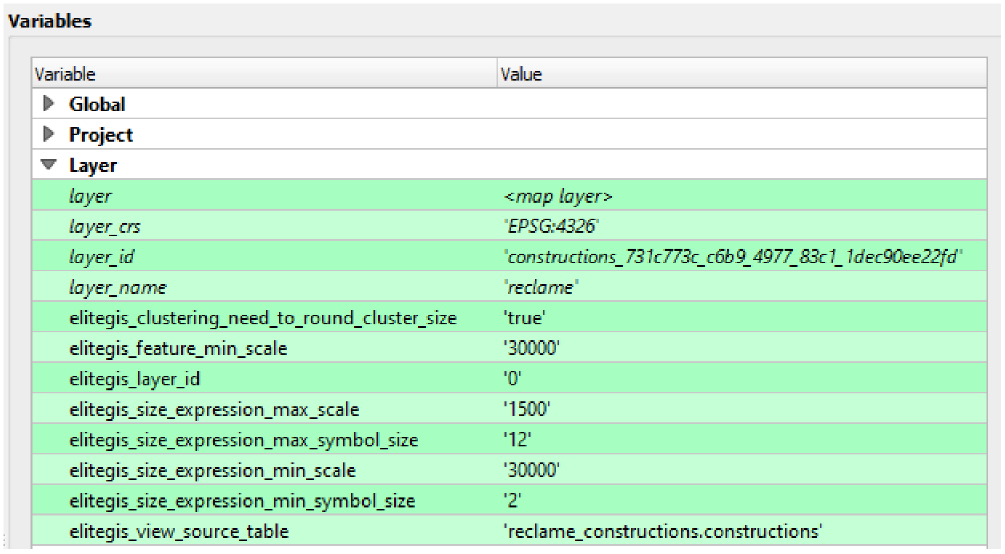
End expression: *"date" + to_interval('1 day')*

Figure 94 - Setting the Start and End Date/Time from Expressions option

- Redraw layer Only – this setting is not supported; it is required only for updating the layer while working with Temporal Controller in QGIS.

5.10. Variables

At the level of single layer in QGIS project it is possible to set variables that will be used by publishing services in **CoGIS Server**, for example: clusterization settings (*Clustering*), scale dependency of symbology (*Scale Dependent*), tracking edits (*Edit Tracker*) and more (see section 6), see Figure 95.


Variables

Variable	Value
Global	
Project	
Layer	
layer	*<map layer>*
layer_crs	*'EPSG:4326'*
layer_id	*'constructions_731c773c_c6b9_4977_83c1_1dec90ee22fd'*
layer_name	*'reclame'*
elitegis_clustering_need_to_round_cluster_size	'true'
elitegis_feature_min_scale	'30000'
elitegis_layer_id	'0'
elitegis_size_expression_max_scale	'1500'
elitegis_size_expression_max_symbol_size	'12'
elitegis_size_expression_min_scale	'30000'
elitegis_size_expression_min_symbol_size	'2'
elitegis_view_source_table	'reclame_constructions.constructions'

Figure 95 – 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 **CoGIS Server**. These variables provide 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 **CoGIS Server** or **CoGIS Portal** levels are required.

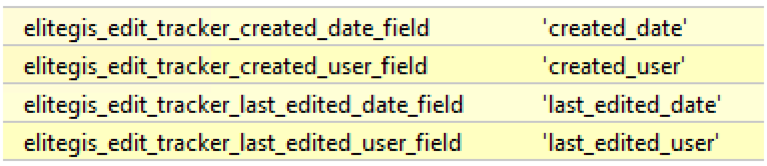
6.1. Tracking edits (Edit Tracker)

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

Table 1 – Edit Tracker settings group

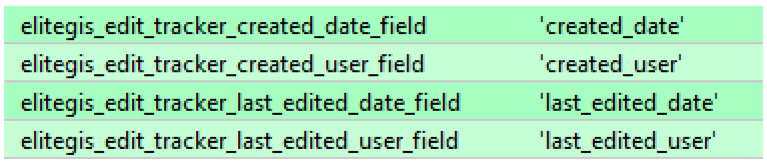
Variable	Description	Properties for
elitegis_edit_tracker_created_date_field	*Field name for completing the date of the project's creation*	*Entire project and layer*
elitegis_edit_tracker_created_user_field	*Field name for completing the name of the user who created the feature*	*Entire project and 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 96) and for the layer (Figure 97).



elitegis_edit_tracker_created_date_field	'created_date'
elitegis_edit_tracker_created_user_field	'created_user'
elitegis_edit_tracker_last_edited_date_field	'last_edited_date'
elitegis_edit_tracker_last_edited_user_field	'last_edited_user'

Figure 96 – Setting Edit Tracker variables for the project



elitegis_edit_tracker_created_date_field	'created_date'
elitegis_edit_tracker_created_user_field	'created_user'
elitegis_edit_tracker_last_edited_date_field	'last_edited_date'
elitegis_edit_tracker_last_edited_user_field	'last_edited_user'

Figure 97 – Setting Edit Tracker variables for the layer

6.2. 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 **CoGIS Server** and by adding this service to the map in **CoGIS Portal**.

6.2.1. 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 98.

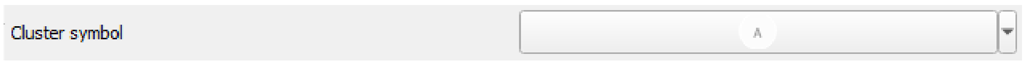

Cluster symbol

Figure 98 – Selecting marker for cluster

The cluster symbol size is set via the Data defined override expression, see Figure 99.

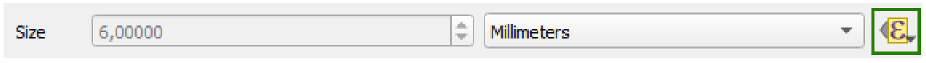

Size 6,00000 Millimeters

Figure 99 – Setting the cluster symbol size (1)

To do so, in the context menu of Data defined override expression select Assistant, see Figure 100.

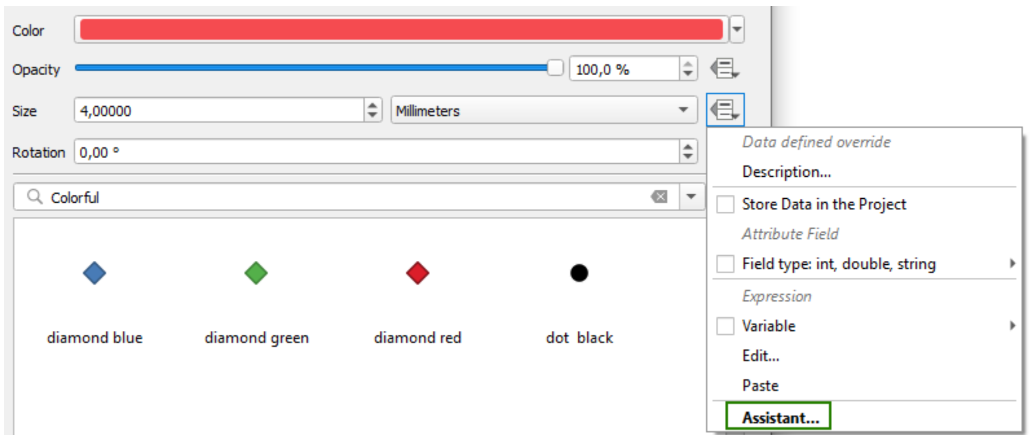
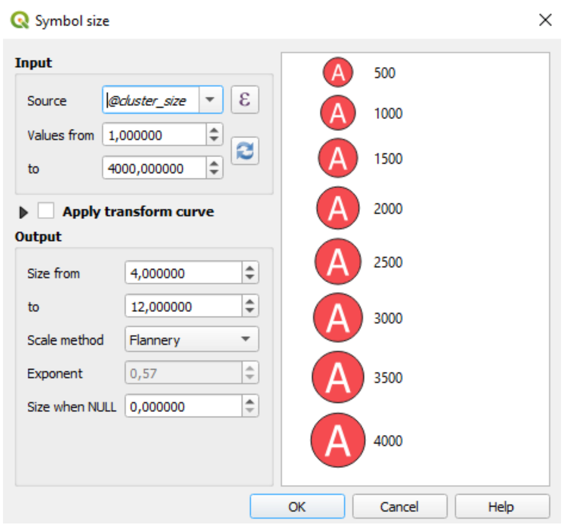

Color
Opacity 100,0 %
Size 4,00000 Millimeters
Rotation 0,00 °
Colorful

Figure 100 – 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 that appeared in the cluster.

The fields *Values from* and *To* show the number of features that can appear to the cluster, see Figure 101.


Symbol size

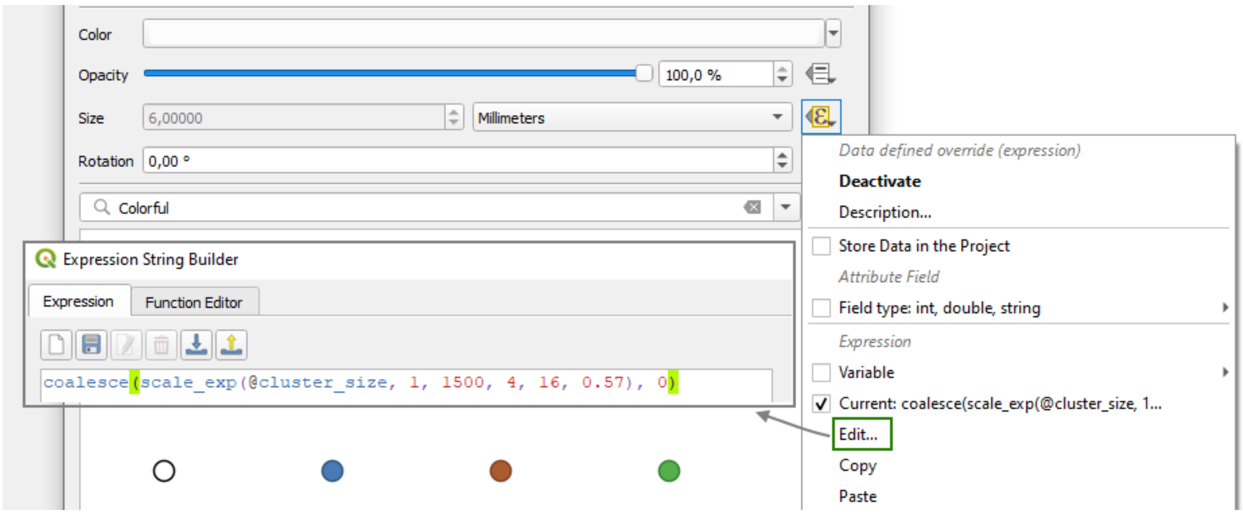
Input
Source @cluster_size
Values from 1,000000
to 4000,000000
Apply transform curve
Output
Size from 4,000000
to 12,000000
Scale method Flannery
Exponent 0,57
Size when NULL 0,000000

Figure 101 – 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 102.


Color
Opacity
100,0 %
Size
6,00000
Millimeters
Rotation
0,00 °
Colorful
Expression String Builder
Expression
Function Editor
Data defined override (expression)
Deactivate
Description...
Store Data in the Project
Attribute Field
Field type: int, double, string
Expression
Variable
Current: coalesce(scale_exp(@cluster_size, 1...
Edit...
Copy
Paste

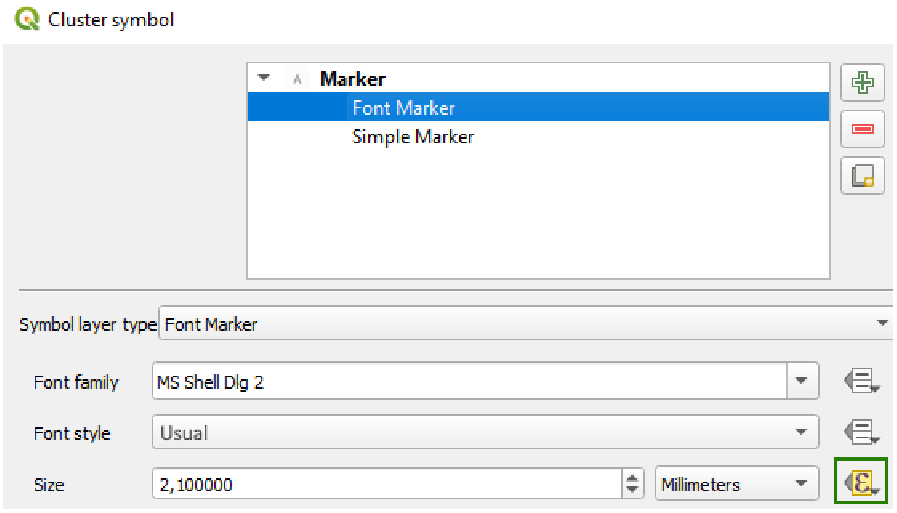
```
coalesce(scale_exp(@cluster_size, 1, 1500, 4, 16, 0.57), 0)
```

Figure 102 – Setting the cluster symbol size (4)

6.2.2. 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 103

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



Symbol layer type: Font Marker
Font family: MS Shell Dlg 2
Font style: Usual
Size: 2,100000 Millimeters

Figure 103 – 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 2.

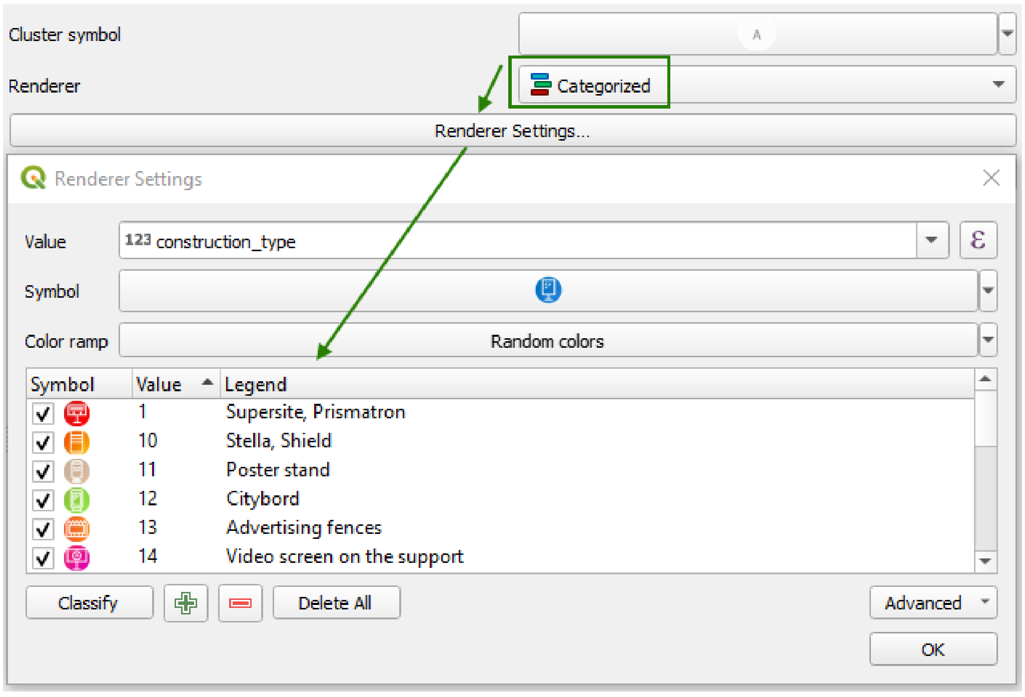
Table 2 – Parameters for additional setting of label

Variable	Description	Value
elitegis_clustering_need_to_scale_labels	*Scaling labels*	*true/false*
elitegis_clustering_show_small_label	*Showing small text*	*true/false*
elitegis_clustering_need_to_round_cluster_size	*Rounding the number of features in the cluster (not more than 4 characters)*	*true/false*

6.2.3. 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 104.



Value: 123 construction_type
Symbol
Color ramp: Random colors

Symbol	Value	Legend
✓	1	Supersite, Prismatron
✓	10	Stella, Shield
✓	11	Poster stand
✓	12	Citybord
✓	13	Advertising fences
✓	14	Video screen on the support

Figure 104 – Rendering the source features of the cluster

6.2.4. Distance

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

6.2.5. 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 3.

Table 3 – Parameters for setting visibility of the source features of the cluster

Variable	Description
elitegis_feature_min_scale	*Minimum scale value*
elitegis_feature_max_scale	*Maximum scale value*

6.2.6. Clusterization type

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

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

6.2.7. Edge symbology

For setting of the edge symbology, the additional layer (temporary layer in QGIS) is used, see Figure 105.

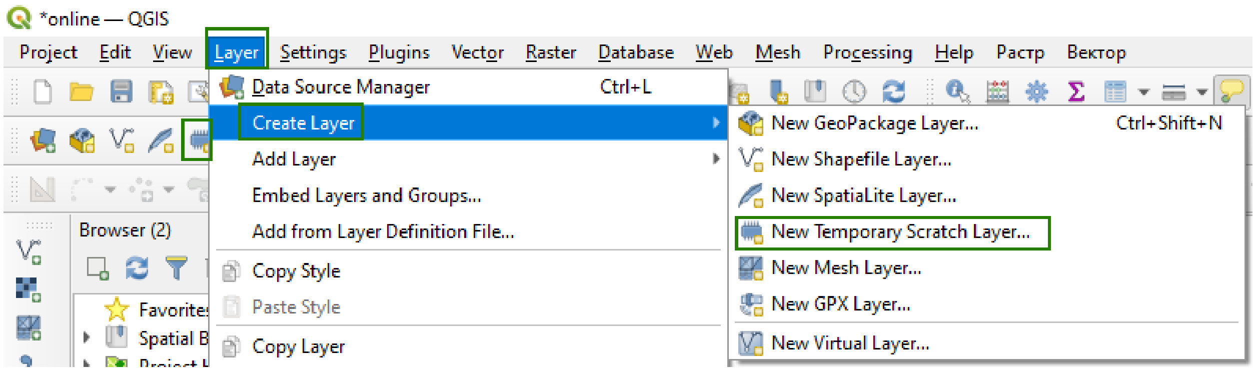
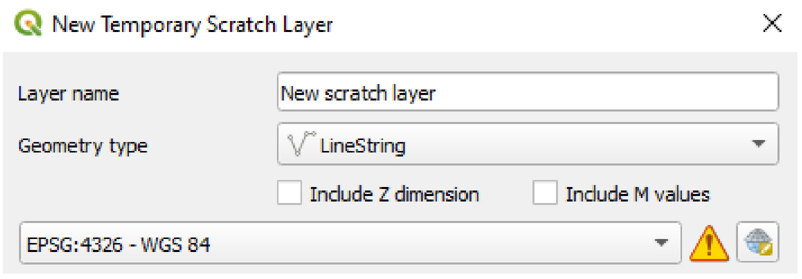


Figure 105 – New temporary layer for cluster edges

The required geometry type of the temporary layer is *LineString*, see Figure 106.



Layer name: New scratch layer
Geometry type: LineString
EPSG:4326 - WGS 84

Figure 106 – Geometry type of the temporary layer with cluster edges

The created temporary layer should be placed just under the main cluster layer, see Figure 107.

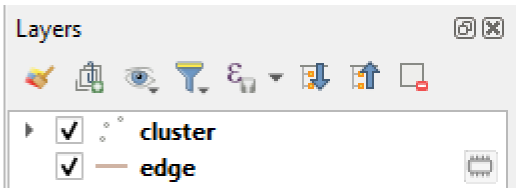


Figure 107 – Location of the layer with cluster edges in the project's layers tree

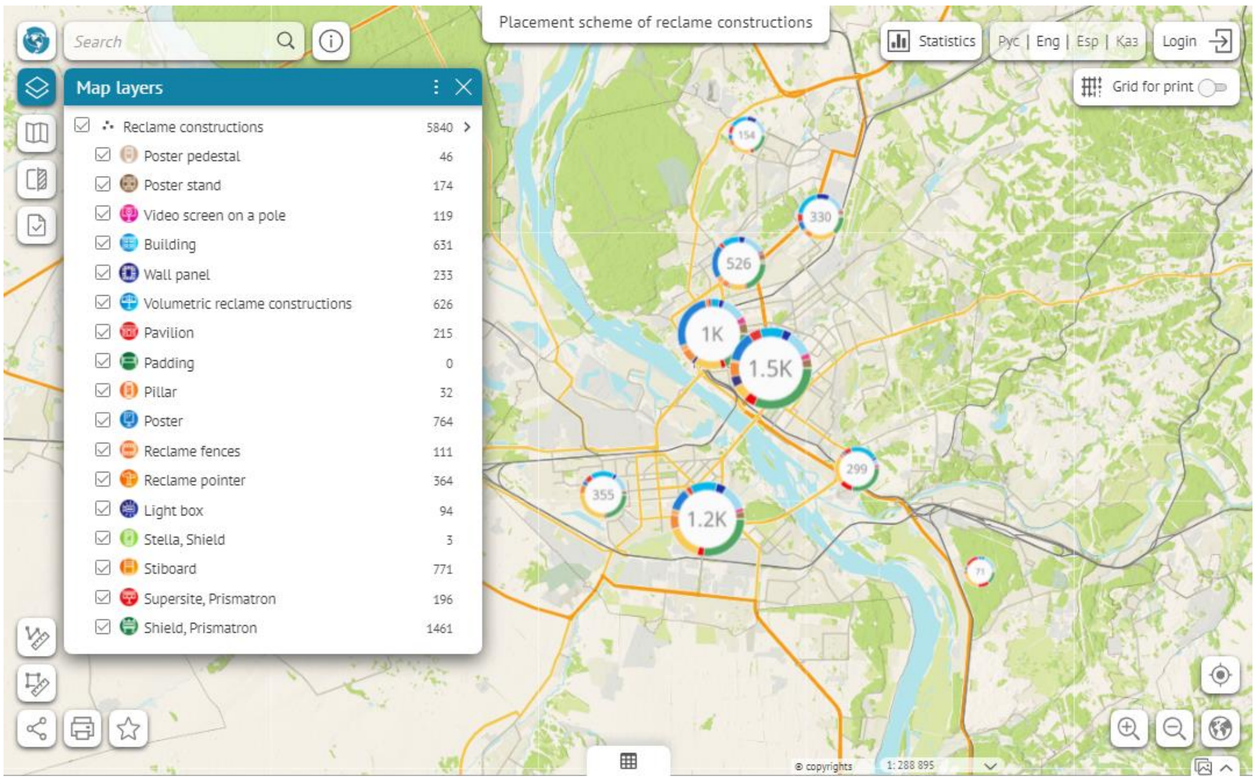
In the layer properties in Variables tab specify parameter *`elitegis_clustering_layer_type`* with value *edge*:

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.

6.2.8. 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 108.



Reclame constructions	5840
Poster pedestal	46
Poster stand	174
Video screen on a pole	119
Building	631
Wall panel	233
Volumetric reclame constructions	626
Pavilion	215
Padding	0
Pillar	32
Poster	764
Reclame fences	111
Reclame pointer	364
Light box	94
Stella, Shield	3
Stiboard	771
Supersite, Prismatron	196
Shield, Prismatron	1461

Figure 108 – 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 109.

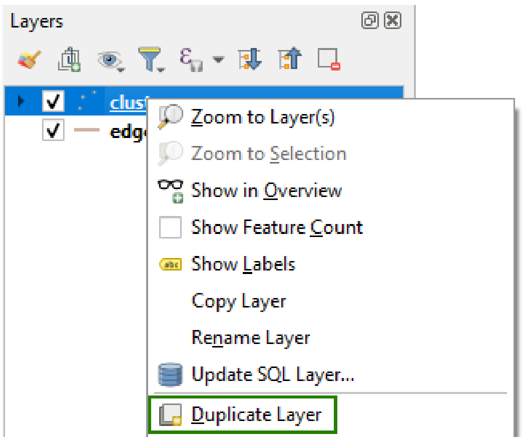


Figure 109 – Duplicating the main cluster layer for creation of the pie chart

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 110.

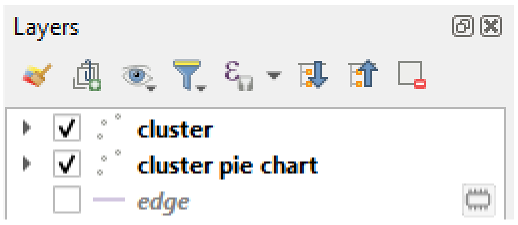


Figure 110 – Location of the setting layer for pie charts relative to the main cluster layer and edges layer

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 ***CoGIS Server*** 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 111.

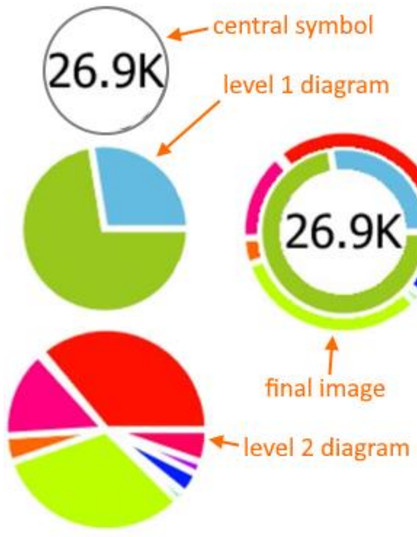
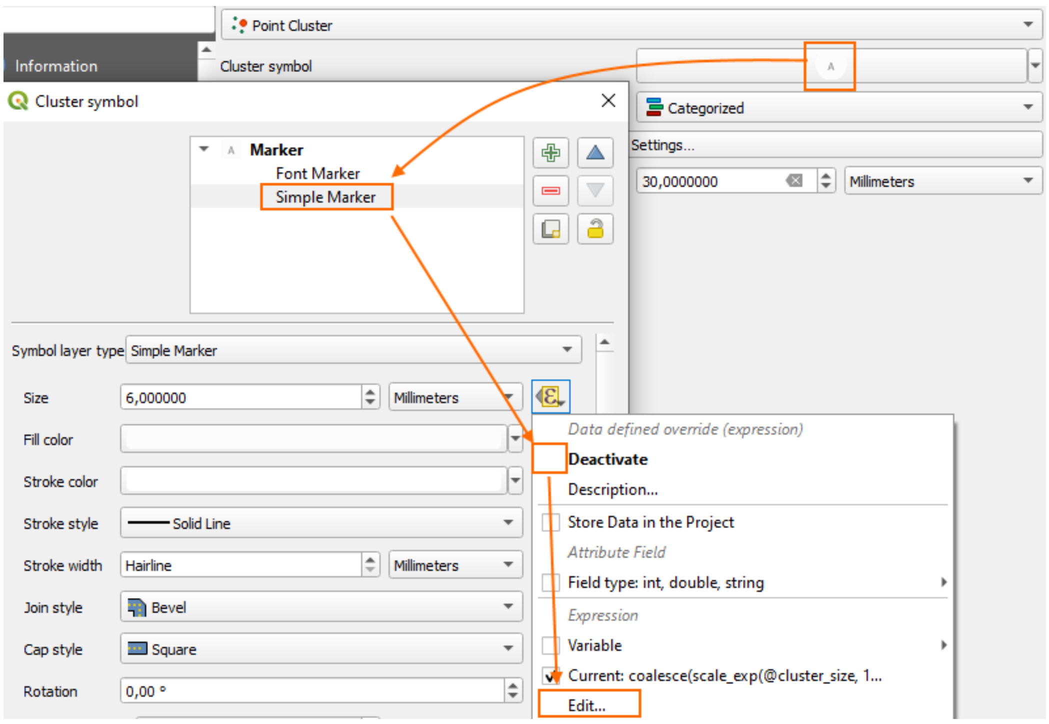


Figure 111 – The order of drawing symbols in the cluster

6.2.8.1. 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 112.



Point Cluster

Cluster symbol: Categorized

Settings...

30,0000000 Millimeters

Symbol layer type: Simple Marker

Size: 6,000000 Millimeters

Fill color

Stroke color

Stroke style: Solid Line

Stroke width: Hairline Millimeters

Join style: Bevel

Cap style: Square

Rotation: 0,00 °

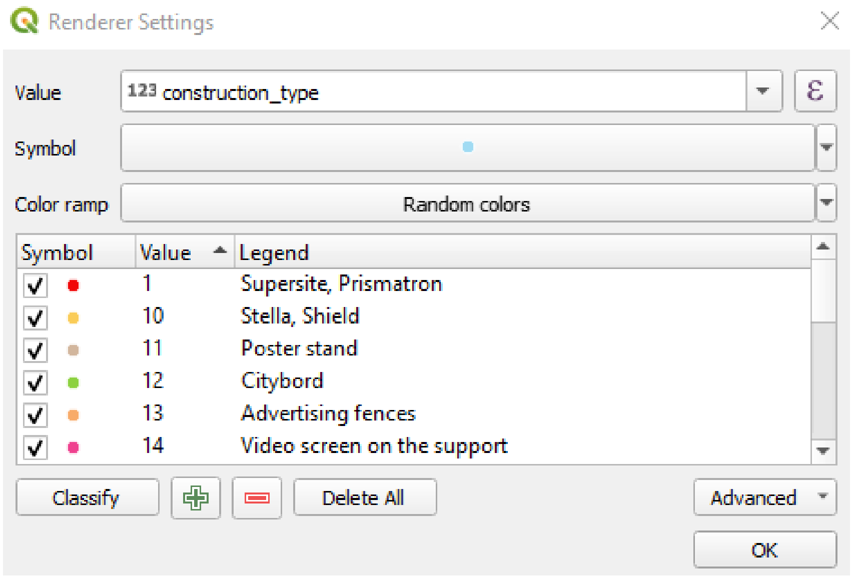
Current: coalesce(scale_exp(@cluster_size, 1...

Figure 112 – 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 113.



Value: 123 construction_type

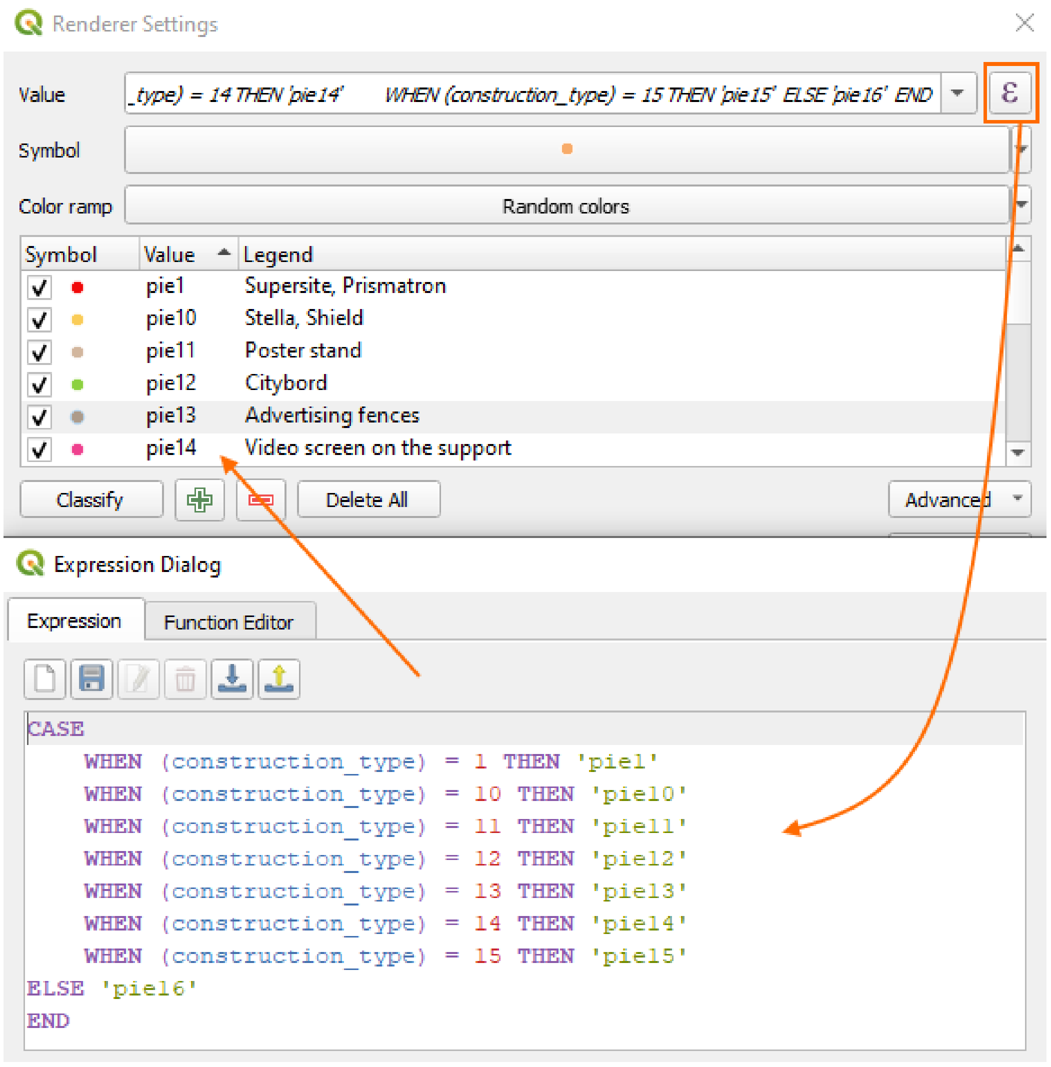
Symbol

Color ramp: Random colors

Symbol	Value	Legend
✓ ●	1	Supersite, Prismatron
✓ ●	10	Stella, Shield
✓ ●	11	Poster stand
✓ ●	12	Citybord
✓ ●	13	Advertising fences
✓ ●	14	Video screen on the support

Figure 113 – 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 114.



Symbol	Value	Legend
✓ ●	pie1	Supersite, Prismatron
✓ ●	pie10	Stella, Shield
✓ ●	pie11	Poster stand
✓ ●	pie12	Citybord
✓ ●	pie13	Advertising fences
✓ ●	pie14	Video screen on the support

```
CASE
    WHEN (construction_type) = 1 THEN 'pie1'
    WHEN (construction_type) = 10 THEN 'pie10'
    WHEN (construction_type) = 11 THEN 'pie11'
    WHEN (construction_type) = 12 THEN 'pie12'
    WHEN (construction_type) = 13 THEN 'pie13'
    WHEN (construction_type) = 14 THEN 'pie14'
    WHEN (construction_type) = 15 THEN 'pie15'
ELSE 'pie16'
END
```

Figure 114 – 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.

6.2.8.2. Pie chart for single cluster

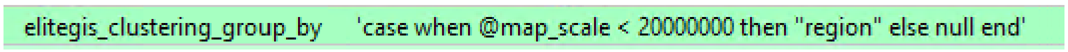
The symbology for displaying pie chart for the single cluster (the feature that did not appear in any cluster) is set via *`elitegis_clustering_single_feature_draw_mode`* parameter.

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.

6.2.9. Grouping clusters by attribute

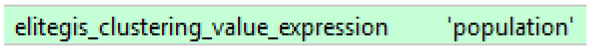
CoGIS Server supports functionality of grouping clusters by value from specific field, that is, to not group features to one cluster it they have different attributes. This option is specified using the following parameter:



elitegis_clustering_group_by	'case when @map_scale < 20000000 then "region" else null end'

For grouping by scale there is an option to specify the scale range within which the grouping is applied. Parameters are *elitegis_clustering_group_min_scale* and *elitegis_clustering_group_max_scale*

6.2.10. Cluster size by the sum of values from the numeric field

By default, the size of the cluster and the value in the label depend on the number of features in the cluster. It is possible to set the cluster size based on the sum of values from the specified field of all features included in the cluster and display this value in the cluster label.
This option is set using the parameter with the value of the field name from which the value to determine the cluster size should be taken: 

elitegis_clustering_value_expression	'population'

6.2.11. Setting cluster identification

This is the option to set the result of the cluster identification. It is set using parameter *elitegis_clustering_identify_mode* with the following values:

- *features* – the list of all features included in the cluster;
- *cluster* – the information about the cluster with the zooming extent;
- *mixed* (by default) – mixed identification type.

For mixed identification type the following additional parameters can be specified:

  - *elitegis_clustering_identify_features_min_scale* - the smallest scale at which, during identification, a list of features included in the cluster will be returned, the default value is 0 (i.e., the extent will always be returned)
  - *elitegis_clustering_identify_features_max_count* - the maximum number of features in the cluster to return a list of features, and not the extent of all features in the cluster, the default value is 1.

6.3.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.

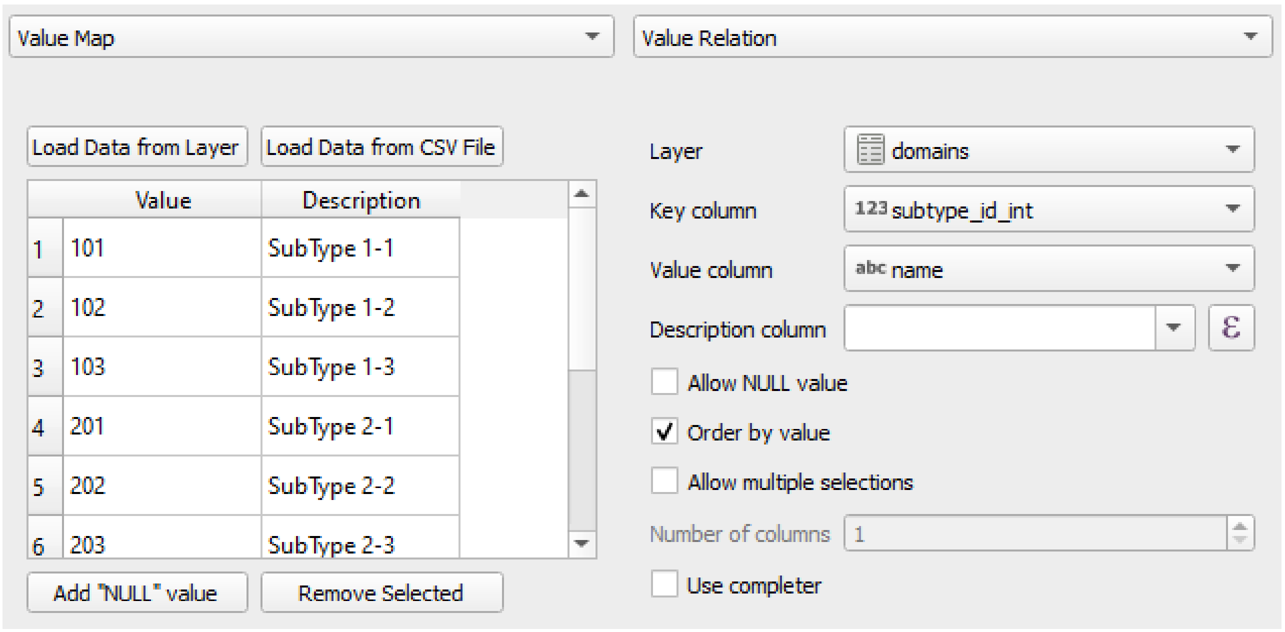
CoGIS Server and CoGIS Portal support subtypes set at the level of the map project in QGIS.

Setting subtypes is done via filtration of specified domains for the child field relative to the value of the parent field.

There can be several child fields. There is only one parent field for one layer. The parent field should be of the integer type (*smallint, integer, bigint*).

The child fields can be of integer (*smallint, integer, bigint*), string (text, *character variyng*) and *uuid* type.
For the child field it is required to specify domains with all possible values and in any way: either via the value map (InPlace domain), or via the related value from another layer (TableBased domain).

The example of setting domain for the child field is shown on Figure 116.



Value Map

Load Data from Layer | Load Data from CSV File

	Value	Description
1	101	SubType 1-1
2	102	SubType 1-2
3	103	SubType 1-3
4	201	SubType 2-1
5	202	SubType 2-2
6	203	SubType 2-3

Add "NULL" value | Remove Selected

Value Relation

Layer	domains
Key column	123 subtype_id_int
Value column	abc name
Description column	
Allow NULL value	☐
Order by value	✓
Allow multiple selections	☐
Number of columns	1
Use completer	☐

Figure 115 – Example of setting domains for the child field

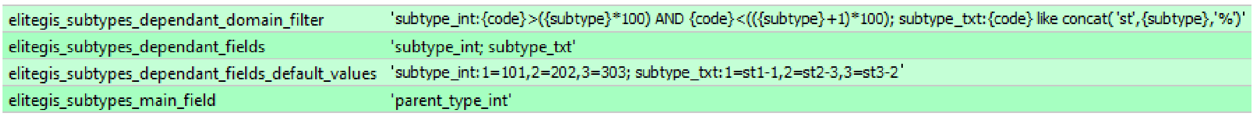
The subtypes settings are specified via specific variables for the layer, see Table 4.

Table 4 – Variables for setting subtypes in layer

Variable	Example of value of one/multiple fields	Description
elitegis_subtypes_main_field	*parent_field*	*Name of the parent field*
elitegis_subtypes_dependant_fields	*child_field*	*Name/names of child fields*
	child_field1; child_field2	
elitegis_subtypes_dependant_fields_default_values	*1=101,2=201,3=301*	*Default values for child field/fields*
	child_field1:1=101,2=201,3=301; child_field2:1=st11,2=st21,3=st31;	
elitegis_subtypes_dependant_domain_filter	*{code}>({subtype}*100) AND {code}<(({subtype}+1)*100)*	*Filter for numeric child InPlace domain*
	{code} like concat('st',{subtype},'%')	*Filter for string child InPlace domain*
	parent_field	*Field name as filter of TableBased domain*
	parent_field = {subtype}	*Explicit filter specification for TableBased domain*
	*child_field1:{code}>({subtype}*100) AND {code}<(({subtype}+1)*100); child_field2:{code} like concat('st',{subtype},'%')*	*Filter specification for multiple child fields*

The default values for text field are specified without quotes, for example: *1=st11*
The default values for uuid field are specified in in curly brackets, for example: *1={0a2f168e-8b9e-49ed-9a7a-edf8d79b5ee9}*
The following macros are used for filtering of the child domain:
{subtype} – the parent field value
{code} – the child field value

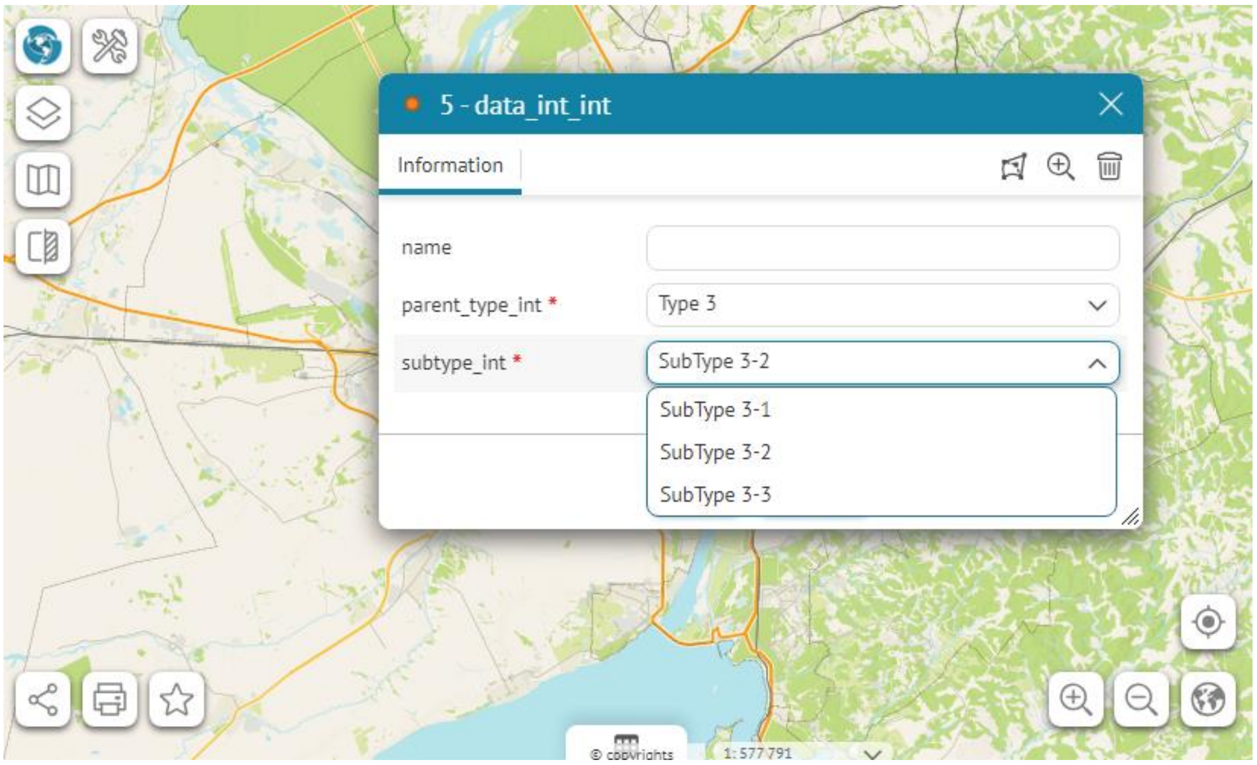
The example of filled variables for the table is shown in the Figure 116.



elitegis_subtypes_dependant_domain_filter	'subtype_int:{code}>({subtype}*100) AND {code}<(({subtype}+1)*100); subtype_txt:{code} like concat('st',{subtype},'%')'
elitegis_subtypes_dependant_fields	'subtype_int; subtype_txt'
elitegis_subtypes_dependant_fields_default_values	'subtype_int:1=101,2=202,3=303; subtype_txt:1=st1-1,2=st2-3,3=st3-2'
elitegis_subtypes_main_field	'parent_type_int'

Figure 116 – Example of parameters set for the subtypes in the layer

The example of the map with features, for one of the attributes of which subtypes are set, is shown on Figure 117. 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*).



5 - data_int_int

Information

name	
parent_type_int *	Type 3
subtype_int *	SubType 3-2

SubType 3-1
SubType 3-2
SubType 3-3

Figure 117 – Creating the new feature, considering the grouping of features by subtypes

6.4.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.

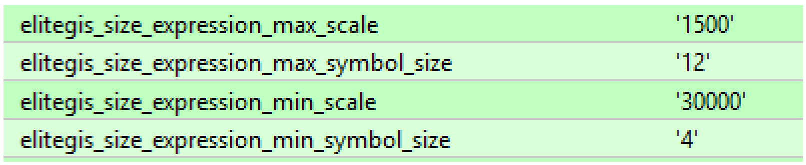
CoGIS Server supports settings of dependency of symbols on the scale, specified at the level of the map project in QGIS.

These settings are specified via variables for the layer, see Table 5.

Table 5 – Variables for setting the semi scale dependency of the layer symbols

Variable	Description	Properties for
elitegis_size_expression_max_scale	*Maximum scale value*	*Layer*
elitegis_size_expression_max_symbol_size	*Increase factor value*	*Layer*
elitegis_size_expression_min_scale	*Minimum scale value*	*Layer*
elitegis_size_expression_min_symbol_size	*Reduction factor value*	*Layer*

The example of filled values for variables is shown on Figure 118.



elitegis_size_expression_max_scale	'1500'
elitegis_size_expression_max_symbol_size	'12'
elitegis_size_expression_min_scale	'30000'
elitegis_size_expression_min_symbol_size	'4'

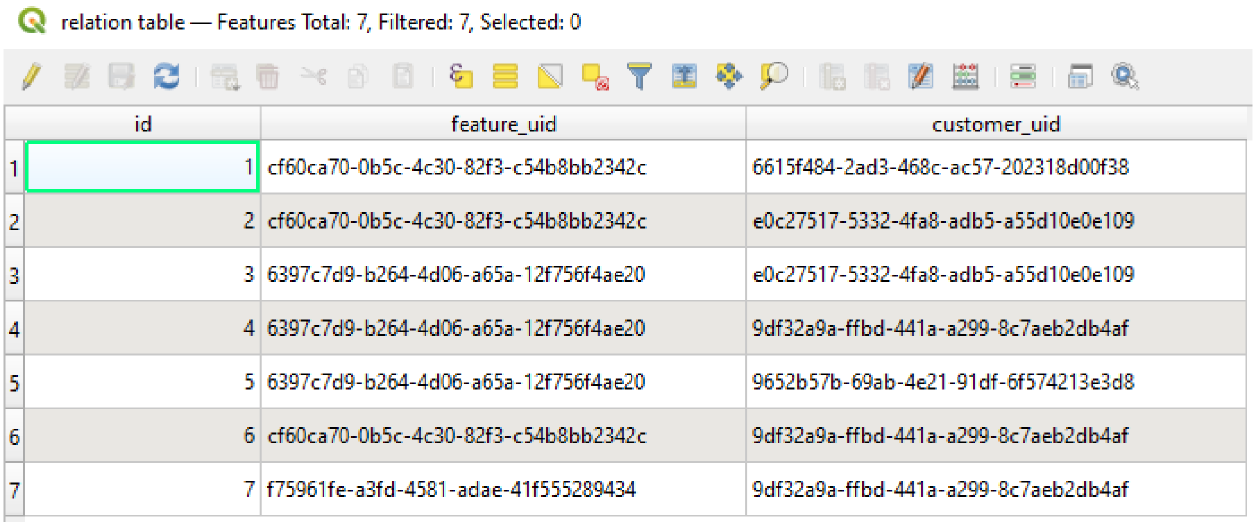
Figure 118 – Example of set variables for the layer display

The elitegis_size_expression parameter allows you to set your own expression for defining the symbol size.

6.5. 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 **CoGIS Server** and **CoGIS Portal**, it is required to add the relationship table to the QGIS project, as shown on Figure 119.



relation table — Features Total: 7, Filtered: 7, Selected: 0

	id	feature_uid	customer_uid
1	1	cf60ca70-0b5c-4c30-82f3-c54b8bb2342c	6615f484-2ad3-468c-ac57-202318d00f38
2	2	cf60ca70-0b5c-4c30-82f3-c54b8bb2342c	e0c27517-5332-4fa8-adb5-a55d10e0e109
3	3	6397c7d9-b264-4d06-a65a-12f756f4ae20	e0c27517-5332-4fa8-adb5-a55d10e0e109
4	4	6397c7d9-b264-4d06-a65a-12f756f4ae20	9df32a9a-ffbd-441a-a299-8c7aeb2db4af
5	5	6397c7d9-b264-4d06-a65a-12f756f4ae20	9652b57b-69ab-4e21-91df-6f574213e3d8
6	6	cf60ca70-0b5c-4c30-82f3-c54b8bb2342c	9df32a9a-ffbd-441a-a299-8c7aeb2db4af
7	7	f75961fe-a3fd-4581-adae-41f555289434	9df32a9a-ffbd-441a-a299-8c7aeb2db4af

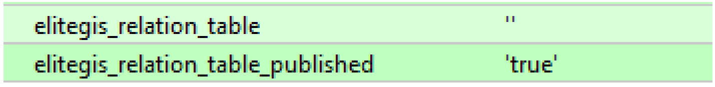
Figure 119 – Example of relationship table for setting the many-to-many relationship at the project's level

Next you need to set variables for this table, as shown in Table 6.

Table 6 – 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)*

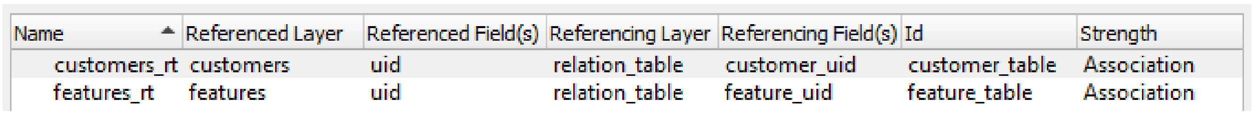
The example of filled values for the relationship table is shown on Figure 120.



elitegis_relation_table	''
elitegis_relation_table_published	'true'

Figure 120 – Example of filled values for the relationship table

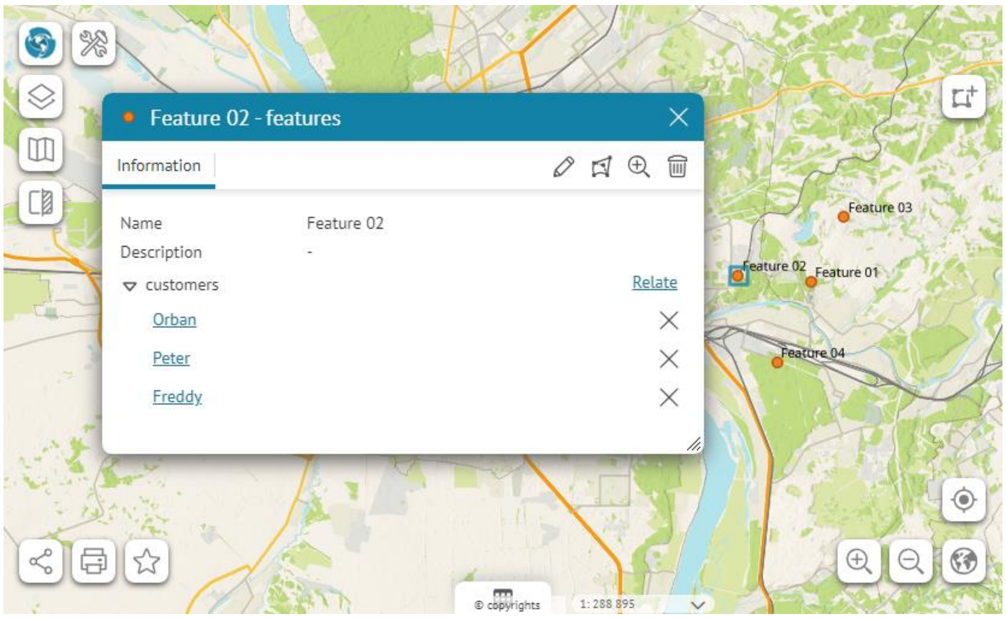
To finish settings, specify the relationship in the project properties for each related layer, see section 3.4 for details. The example of filled data about relationships in the project properties is shown on Figure 121.



Name	Referenced Layer	Referenced Field(s)	Referencing Layer	Referencing Field(s)	Id	Strength
customers_rt	customers	uid	relation_table	customer_uid	customer_table	Association
features_rt	features	uid	relation_table	feature_uid	feature_table	Association

Figure 121 – 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 122-Figure 124.



Feature 02 - features

Information

Name	Feature 02
Description	-

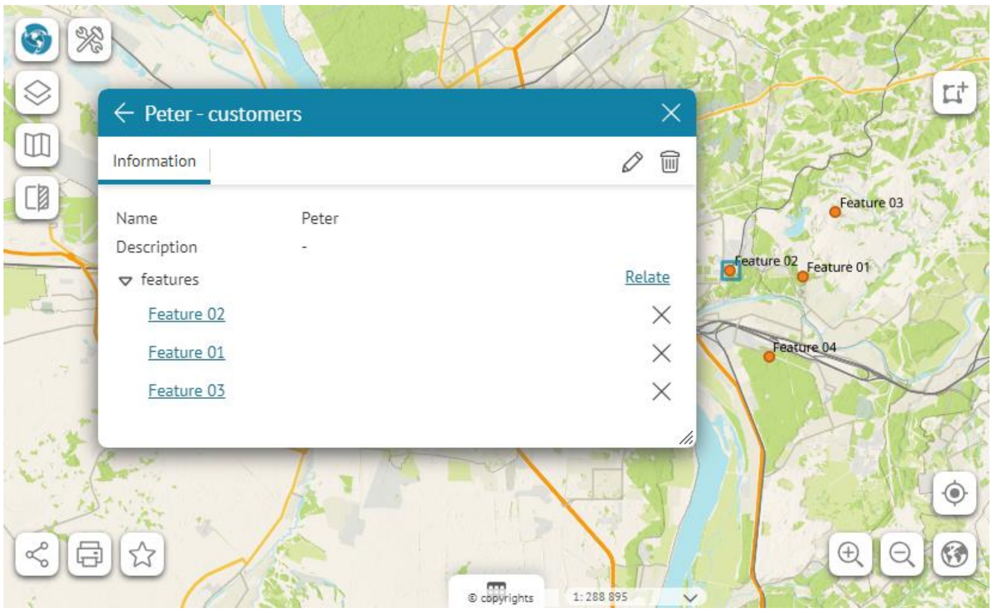
customers Relate

Orban

Peter

Freddy

Figure 122 – Example of a feature related with many customers



Peter - customers

Information

Name	Peter
Description	-

features Relate

Feature 02

Feature 01

Feature 03

Figure 123 – Example of a customer related with many features

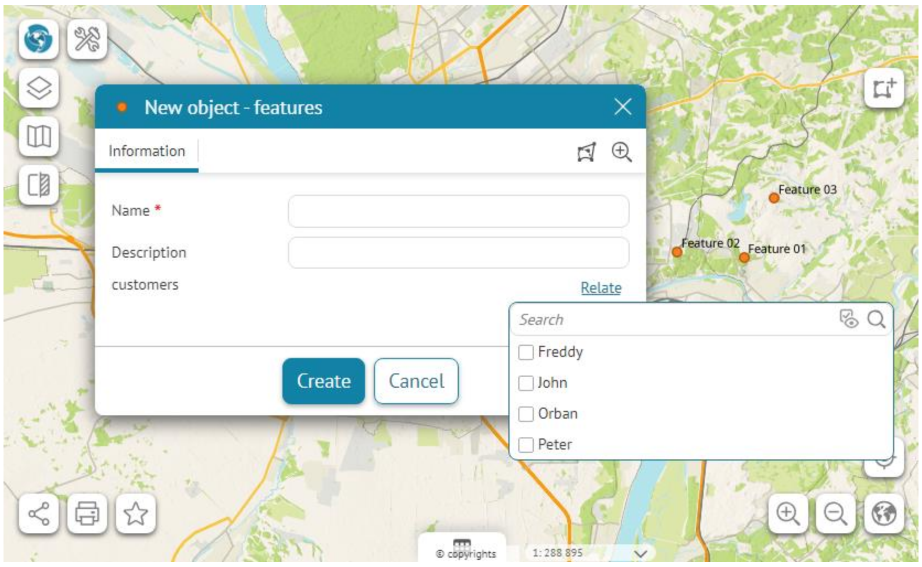
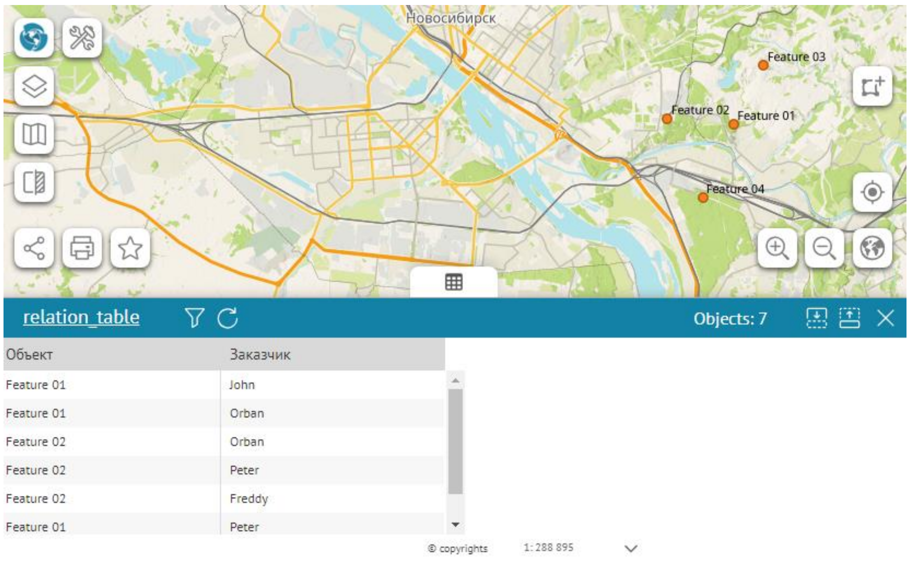


Figure 124 – Creation of the new feature with the option to relate it with one or multiple customers

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 125.



Объект	Заказчик
Feature 01	John
Feature 01	Orban
Feature 02	Orban
Feature 02	Peter
Feature 02	Freddy
Feature 01	Peter

Figure 125 – Relationship table in the published map application

6.6. Annotations

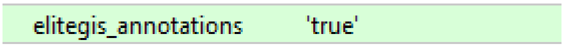
Enabling annotation support for a layer allows you to label polyline features as static labels. All properties for annotations are taken from the properties of the regular label, see section 5.3.

To enable annotation support at the level of **CoGIS Server**, you need to specify the appropriate value for the layer variable, see Table 7.

Table 7 – 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 127.



elitegis_annotations 'true'

Figure 126 – Example of filled variable value for annotation support

6.7.SQL query-based view

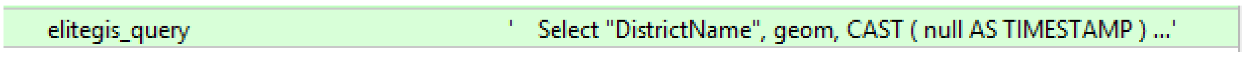
The SQL query-based view is the ability to create layers in a project based on a SQL query to tables in a database. The property is set for the temporary layer.

To enable this option at the **CoGIS Server** level, it is necessary to specify the corresponding value for the variable for the temporary layer in the map project, see Table 8.

Table 8 – 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 127.



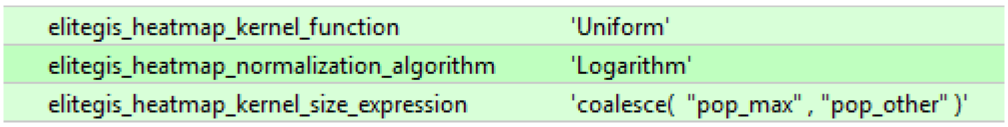
elitegis_query ' Select "DistrictName", geom, CAST (null AS TIMESTAMP) ...'

Figure 127 – Example of filled value for creation of the SQL query-based view

6.8.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 **CoGIS Server**, see Figure 128. At that, no additional service setting at the level of **CoGIS Server** is required.



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

Figure 128 – 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 9.

Table 9 – 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 10 – 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 129.

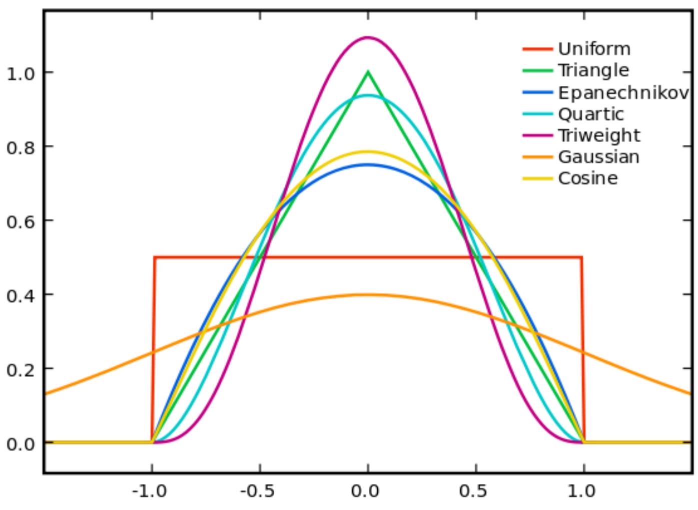


Figure 129 – Graphs of kernel functions for use by rendering of interpolation map

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 11 – 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*
hexagons	*Heatmap mode with hexagons*

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.

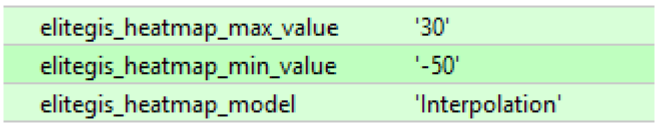
Additionally, the interpolation map can be set by specifying value for normalization window, see Table 12.

Table 12 – 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 130.



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

Figure 130 – Example of filled variables values for the normalization window and the interpolation algorithm

The example of interpolation map published using **CoGIS Server** and **CoGIS Portal** based on the map project described above, is shown on Figure 131.

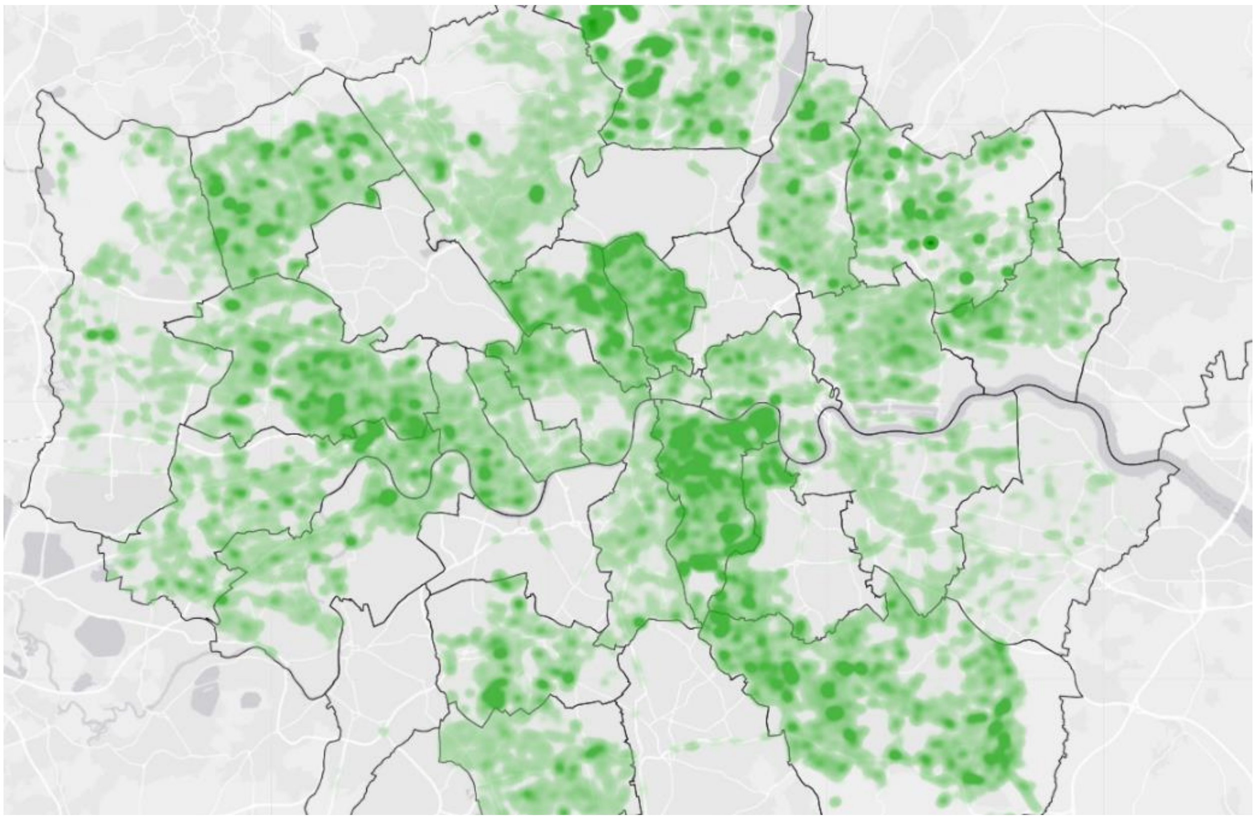


Figure 131 – Interpolation map showing distribution of trees in London districts

6.9.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 132 and Figure 133 show heat maps on air temperature and precipitation in the territory of Russian Federation.

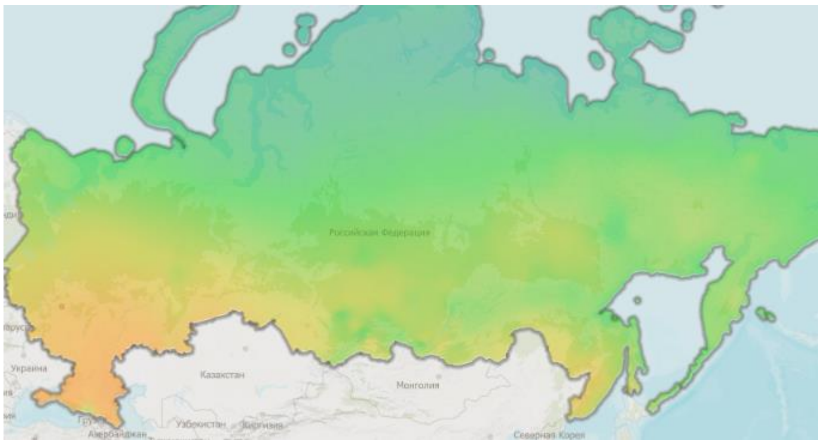


Figure 132 – Heat map on air temperature for specific period in the territory of Russian Federation

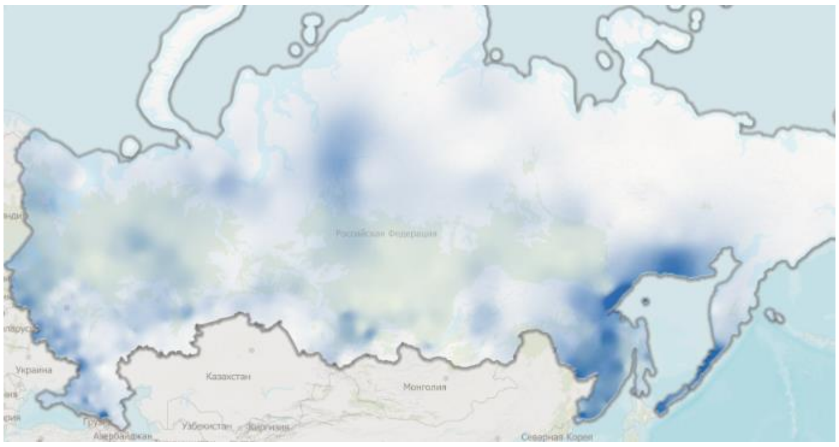


Figure 133 – Heat map on precipitation for specific period in the territory of Russian Federation

The heat maps were created based on weather station readings (see Figure 134) and interpolation of obtained values to the entire territory.

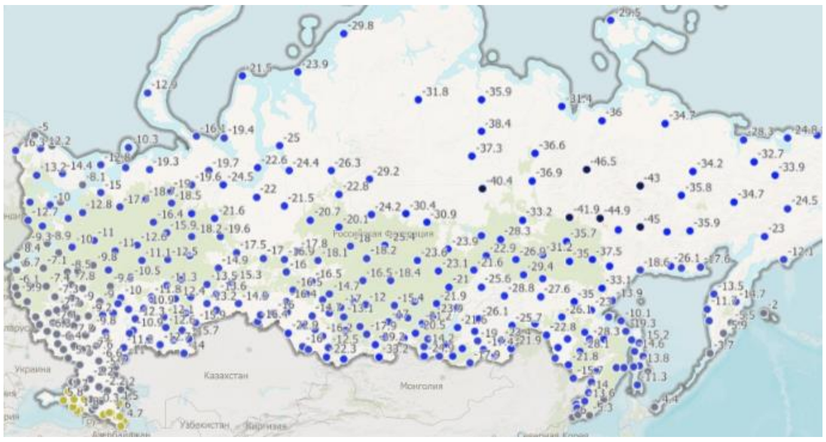


Figure 134 – Source data from weather station based on which the heat maps were created

The heat map can be displayed as hexagons (see Figure 135)

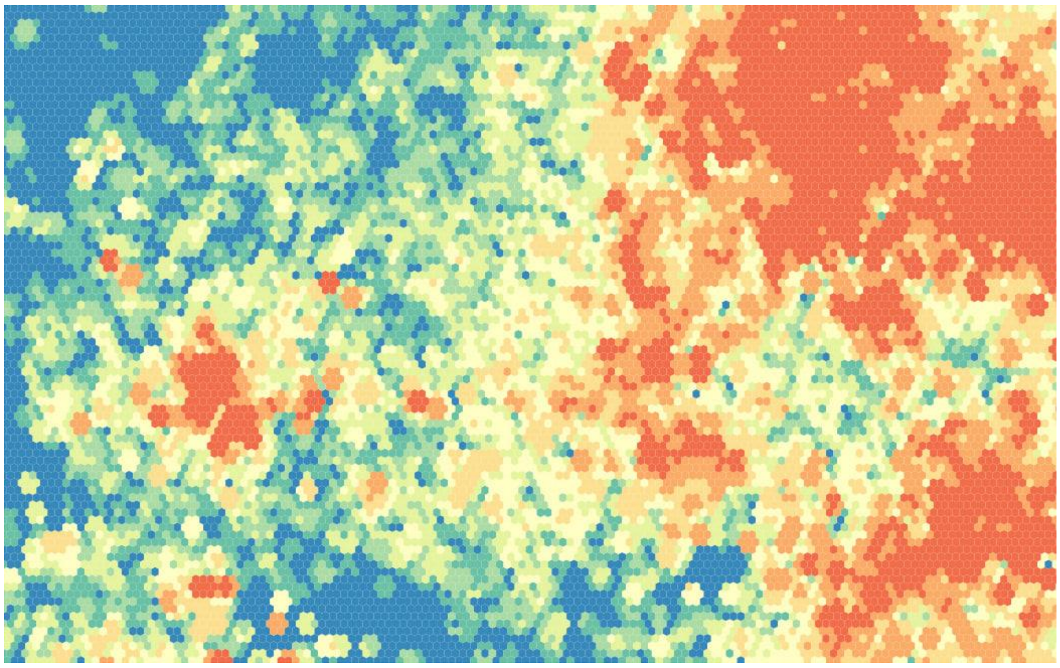


Figure 135 - Heat map in the form of hexagon display

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 **CoGIS Server**:

- `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 **CoGIS Server** 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 13.

Table 13 – 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*
hexagons	*Heatmap mode with hexagons*

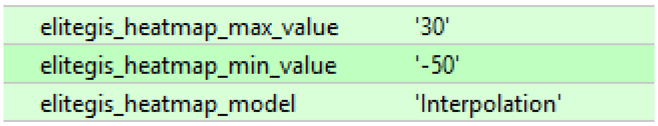
Additionally, the heat map can be set by specifying value for normalization window, see Table 14.

Table 14 – 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 136.



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

Figure 136 – Example of filled variables values for the normalization window and the interpolation algorithm

Examples of heat maps published using **CoGIS Server** and **CoGIS Portal** based on the above-mentioned map project are shown on Figure 132 and Figure 133.

6.10. 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.

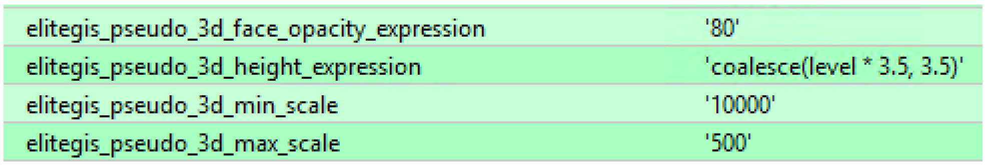
At the level of the selected layer in the QGS project, you can set variables that will allow you to display layer objects as pseudo 3D features when publishing a project in **CoGIS Server**, see Figure 128. At that, no additional setting of service at **CoGIS Server** level is required.

Table 15 – Variables for display of features as pseudo 3D shapes

Variable	Description

elitegis_pseudo_3d_height_expression	Feature's height in meters, *sql expression is allowed*
elitegis_pseudo_3d_face_opacity_expression	*Opacity of side faces in percent (0-100), calculated from the feature's opacity value specified in the symbol coloring*
elitegis_pseudo_3d_min_scale	*Minimum visibility scale of pseudo 3D for the feature (optional)*
elitegis_pseudo_3d_max_scale	*Maximum visibility scale of pseudo 3D for the feature (optional)*

The example of filled values for variables is shown on Figure 137.



elitegis_pseudo_3d_face_opacity_expression	'80'
elitegis_pseudo_3d_height_expression	'coalesce(level * 3.5, 3.5)'
elitegis_pseudo_3d_min_scale	'10000'
elitegis_pseudo_3d_max_scale	'500'

Figure 137 – Example of filled values for variables

The example of pseudo 3D visualization based on a service published with **CoGIS Server** and **CoGIS Portal** and configured at the map project level as described above is shown on Figure 138.

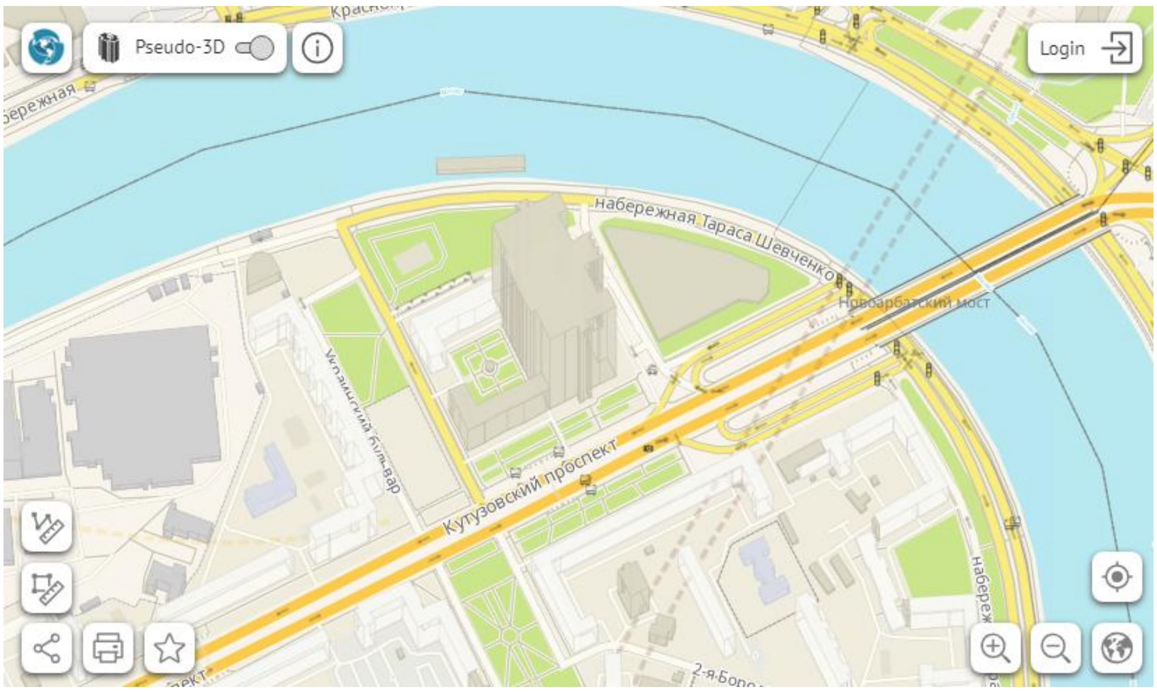


Figure 138 – Example of pseudo 3D visualization

6.11. 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 **CoGIS Server** instead of the automatically assigned ID.

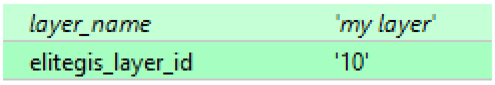
This property can be useful when debugging of QGS project, based on which the map service is published and maps are created in **CoGIS Portal**, as the order, quantity, and IDs of layers, respectively, can change, but at that used in the settings of map in **CoGIS Portal**.

The layer identifier is set via variable *`elitegis_layer_id`*.

At that time 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 139.



layer_name	*'my layer'*
elitegis_layer_id	'10'

Figure 139 – 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: *100=My Group Layer.*

The example of group layer with specified number in QGS project is shown on Figure 140.

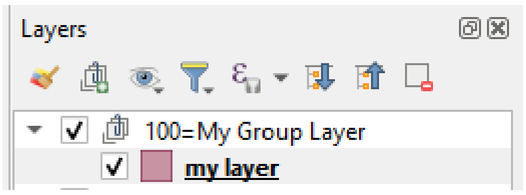


Figure 140 – Example of group layer with specified number

6.12. Cropping map using mask layer

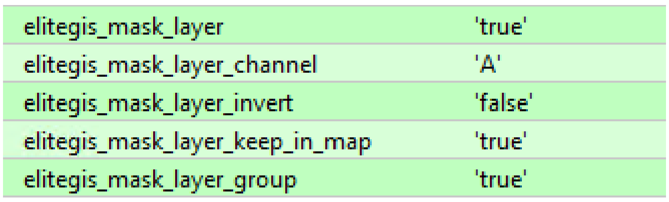
At the level of the QGS project you can configure cropping of all layers of the map service with a special layer called a mask (MaskLayer). The mask layer can be of any type (point, polyline, polygon) and of any color.

The mask layer can be any layer of the map project. The following layer variables are used for setting the mask layer, see Table 15.

Table 16 – Variables for setting mask layer

Variable	Description	Values
elitegis_mask_layer	*Indicates whether the layer is the mask layer*	*true/false*
elitegis_mask_layer_channel	*Indicates which channel should be used as the mask (in most cases, the A opacity is used)*	***A (opacity)*** *R (red)* *G (green)* *B (blue)*
elitegis_mask_layer_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_mask_layer_invert	*Indicates whether the mask inversion should be used*	*true/**false***
elitegis_mask_layer_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 141.



elitegis_mask_layer	'true'
elitegis_mask_layer_channel	'A'
elitegis_mask_layer_invert	'false'
elitegis_mask_layer_keep_in_map	'true'
elitegis_mask_layer_group	'true'

Figure 141 – Example of filled values for setting the mask layer

6.13. 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 142.

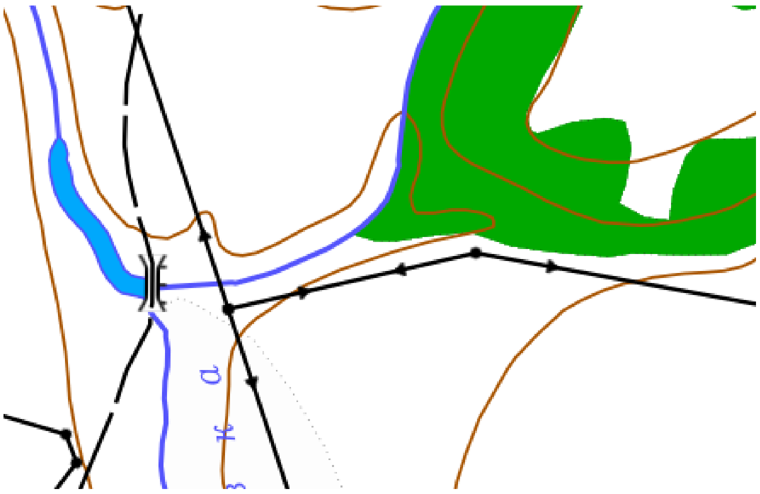


Figure 142 – Example of displaying power line on the topographic map

For setting such symbol in QGIS you need to create a combined linear symbol as shown on Figure 143.

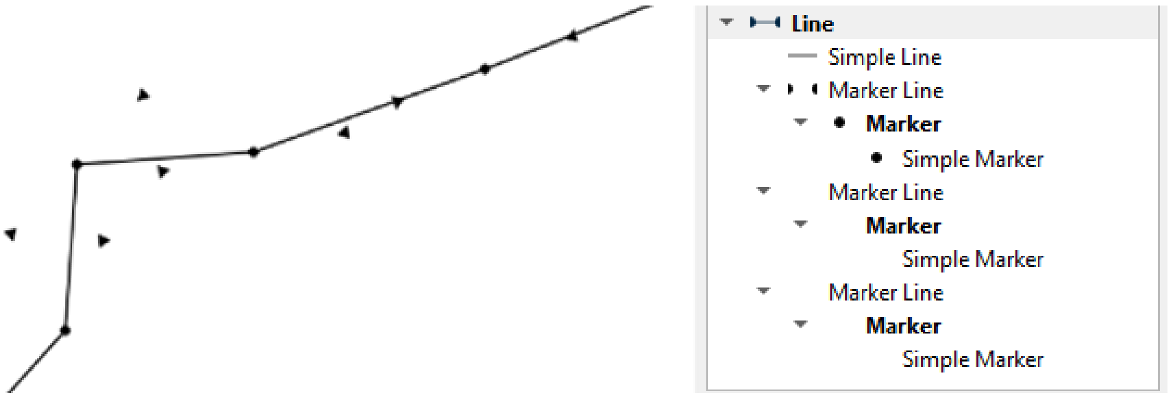
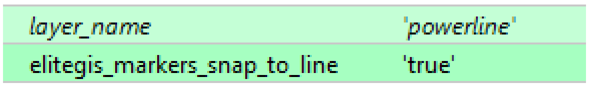


Figure 143 – Setting combined symbol for power line

In the layer parameters specify the variable ***`elitegis_markers_snap_to_line`*** as *true*, see Figure 144.



layer_name	*'powerline'*
elitegis_markers_snap_to_line	'true'

Figure 144 – 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.

6.14. Replacement of data source for display

CoGIS Server provides functionality to use **VIEW** as the data source for display, to edit the source feature class based on which **VIEW** has been created.
For example, there is the feature class based on which VIEW is created and which differs from the source feature class in one or several fields only (calculated values). VIEW will be used for display, but the source feature class will be edited and only those fields that are in the source feature class.
To do so, specify the created VIEW as the source data for the required layer. Specify *elitegis_view_source_table* property in the variables and specify the source table, see Figure 145.

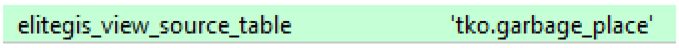

elitegis_view_source_table 'tko.garbage_place'

Figure 145 – Example of a filled variable value that allows you to replace the data source for display

This functionality allows you to avoid the creation of additional triggers for copying values between the related features (from the parent feature to the child feature, etc.), and just to add the required fields to the related feature for display.

6.15. Layer filter with ability to use macros

CoGIS Server supports functionality to specify filter for layer via the variable in the QGIS project. Similar functionality is described in sections 5.1.3 and 5.1.4, the difference is in more flexible sql expression for the filter and different set of macros in the expression.
In the variables specify property *elitegis_where_clause* in values of which specify the needed sql expression with the filter, i.e. the part after **WHERE** only. This part is added to all the remained filters via the logical **AND**. The example of filled variable is shown on Figure Figure ***146***6.
The set of supported macros is shown in Table 17.

Table 17 – Macros for substitution in sql filter for layer

Macros	Value
{CurrentUser}	Login of the current user or empty string for the not authorized user
{CurrentGroups}	All groups where user is included, separated by semicolons, and surrounded by semicolons (for example ;g1;g2;) or empty string for the not authorized user or the user not included in any group
{CurrentGroup.g1}	Whether current user is included to specified group g1 (1 - yes, 0 - no)
{IsUserLoggedIn}	Whether user is authorized or not authorized (1 – authorized, 0 - anonymous)
{CurrentUserFullName}	Name of authorized user empty string for the not authorized user
{WhereClause}	External filter for layer, like property *elitegis_query*

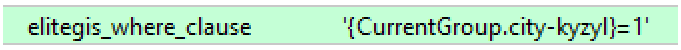

elitegis_where_clause '{CurrentGroup.city-kyzyl}=1'

Figure 146 – Example of filled variable of filter for layer

6.16. 3D diagrams

CoGIS Server supports display of 3D diagrams (see section 5.4).

For the layer in which the diagrams display is specified, specify variable `elitegis_chart_symbol_3d` as `true`, see Figure 147 and Figure 148.

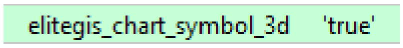

elitegis_chart_symbol_3d 'true'

Figure 147 – Example of filled variable provided for display of 3D diagrams

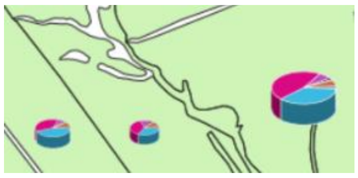


Figure 148 – Example of 3D diagram display

6.17. Sorting layer features by field to return in the request

CoGIS Server supports the ability to determine the sequence of returning layer features by the request to the layer.
In the variables specify *elitegis_features_order_by* property and in this property specify the field name and the sorting method ***ASC/DESC***, see Figure 149.

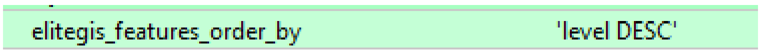

elitegis_features_order_by 'level DESC'

Figure 149 – Example of filled variable for sorting features in the request to layer

6.18. Excluding layer from map service

CoGIS Server supports the ability to exclude specific layer from the published map service without deleting this layer from the QGS project.

In the variables specify *elitegis_ignore* property and in this property specify **true/false**, see Figure 150

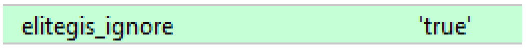

elitegis_ignore 'true'

Figure 150 - Example of filled variable for excluding layer from map service

This option can be applied for reference books used in the domains of other project layers. By excluding the reference book, the related values based on this reference book remain.

6.19 Displaying the visible area considering the relief

The setting highlights the area visible to the observer, considering the height map relative to the ground set by the observer.

For QGIS this will be a regular point layer with the following symbology types: Simple symbol, Unique values, Rules, Graduated symbol. Though each symbol should be simple, that is the symbol determining its own highlight color of the visible area, the other symbol settings are ignored.

Besides, the raster layer with heights (relief) should be added to the project.

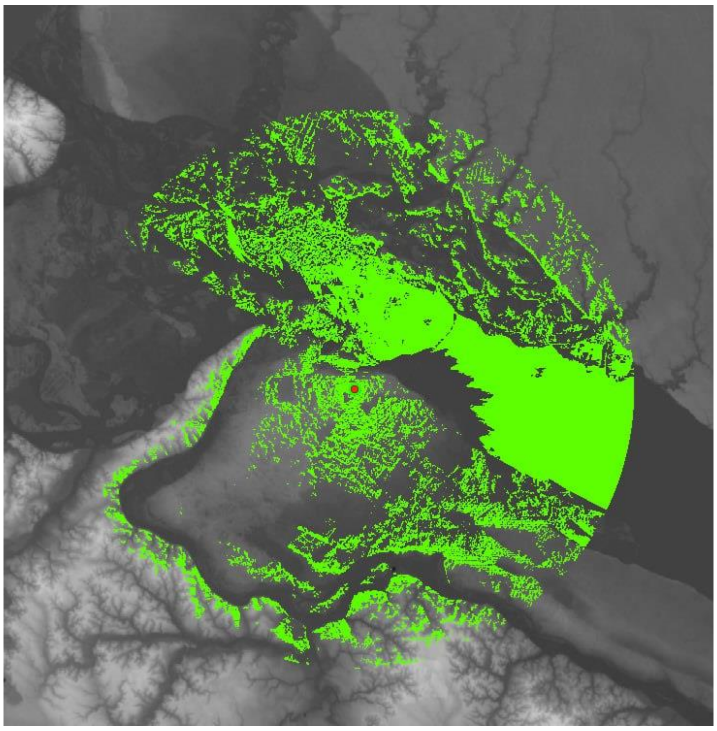


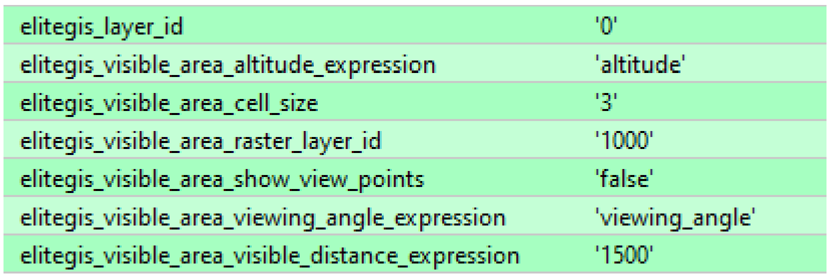
Figure 151 – Example of display of visible are considering the relief

Available setting variables are shown in Table 18.

Table 18 – Variables for setting the display of visible are considering the relief

Macros	Value	Mandatory/ By default
elitegis_visible_area_raster_layer_id	Number of raster layer with heights in the same QGS project (elitegis_layer_id equal to this value)	Yes
elitegis_visible_area_altitude_expression	The height of the observer in meters, sql expression that uses fields of the source point layer of the observer or the constant	No By default 3m
elitegis_visible_area_cell_size	Number of pixels per cell (the more is the value, the worse is the area display quality)	No By default 1
elitegis_visible_area_viewing_angle_expression	Viewing angle in n degrees; if this angle is set, then the direction of view will be affected by the angle of rotation of the symbol from the symbol drawing settings in QGS project	No By default 360°
elitegis_visible_area_show_view_points	Logical variable that determines whether to draw points of the observer location	No By default true

elitegis_visible_area_visible_distance_expression	Visibility distance of the observer, i.e. this parameter limits the viewing radius from the point of view	No By default is limited by horizon



elitegis_layer_id	'0'
elitegis_visible_area_altitude_expression	'altitude'
elitegis_visible_area_cell_size	'3'
elitegis_visible_area_raster_layer_id	'1000'
elitegis_visible_area_show_view_points	'false'
elitegis_visible_area_viewing_angle_expression	'viewing_angle'
elitegis_visible_area_visible_distance_expression	'1500'

Figure 152 – Example of setting the visibility area

7. Creating geocoding service

CoGIS Server allows you to create geocoding service by map service.

In general case, the geocoding service in **CoGIS Server** 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
- the index for the selected fields is built
- the map project is created from the selected layers

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.

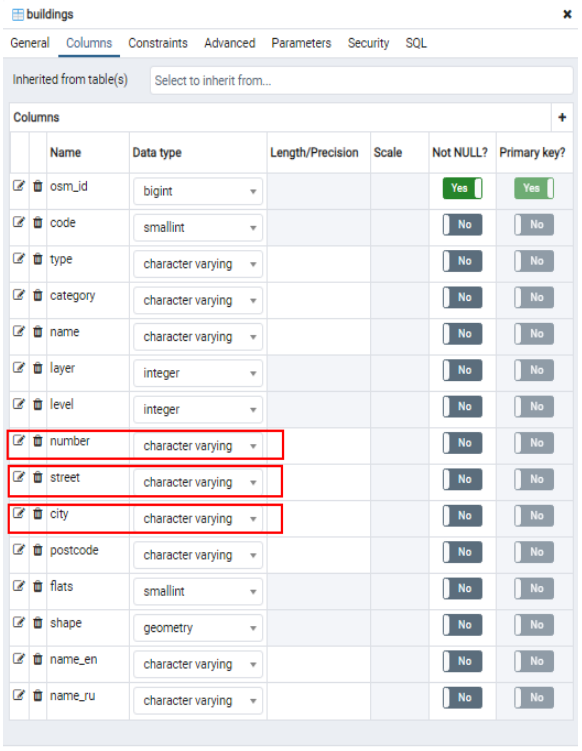
7.1. Preparing data in database

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 the 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 153). In the roads layer the field *name* will be used (see Figure 154).

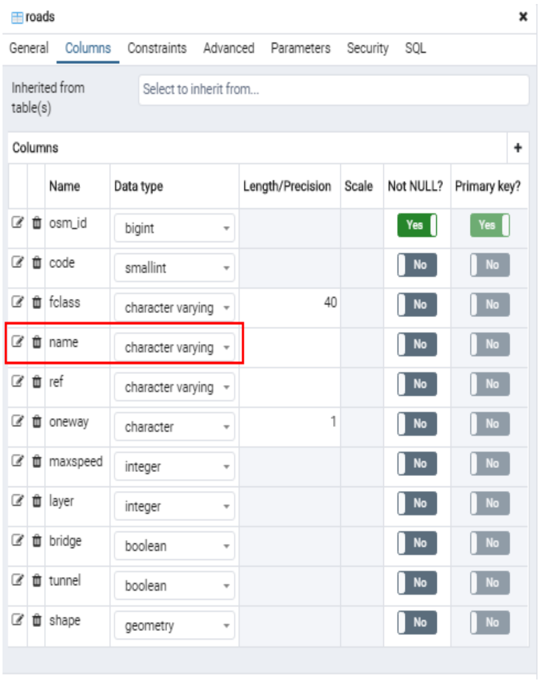


Inherited from table(s): Select to inherit from...

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 153 – 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

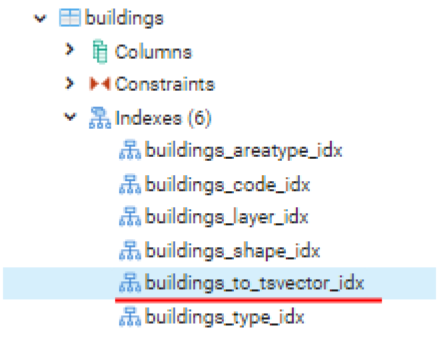
Figure 154 – Roads layer: fields for geocoding

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 155.



```
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 155 – Buildings layer: indices for geocoding

7.2. Preparing the project in QGIS

For publishing of geocoding service in **CoGIS Server** 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 156.

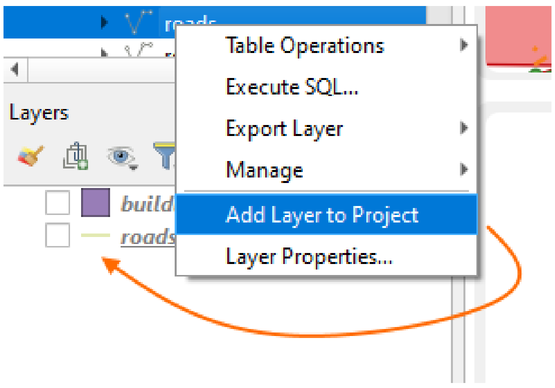


Figure 156 – Adding layers for geocoder to the QGIS map project

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 157.

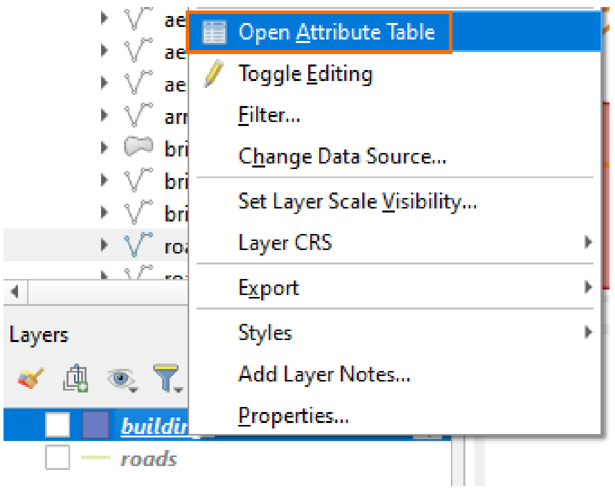
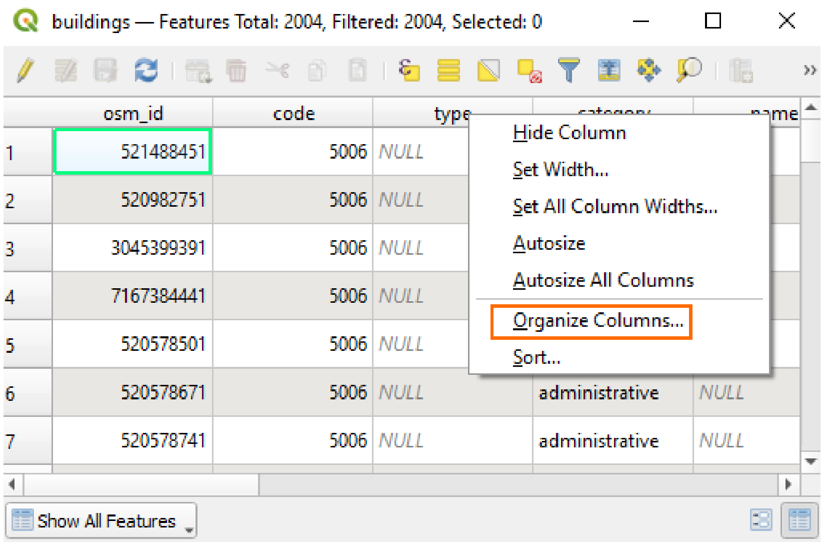


Figure 157 – Attribute table of the layer

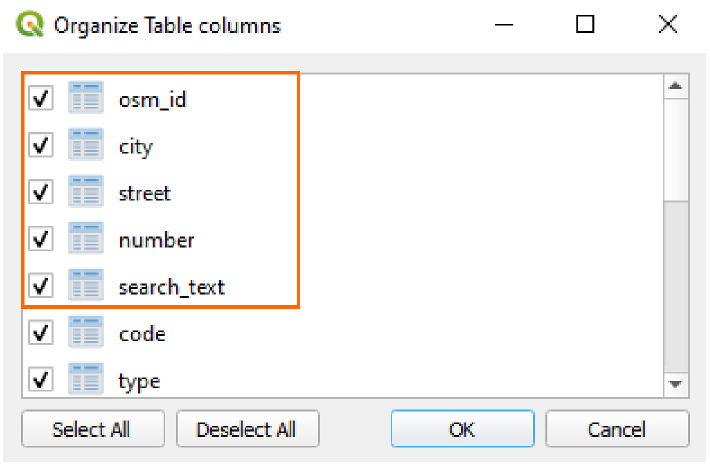
Next, in the attribute table, from the context menu for the column headings, select *Organize Columns*, see Figure 158.


buildings — Features Total: 2004, Filtered: 2004, Selected: 0

	osm_id	code	type	category	name
1	521488451	5006	NULL		
2	520982751	5006	NULL		
3	3045399391	5006	NULL		
4	7167384441	5006	NULL		
5	520578501	5006	NULL		
6	520578671	5006	NULL	administrative	NULL
7	520578741	5006	NULL	administrative	NULL

Figure 158 – 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 159. Press *OK*.

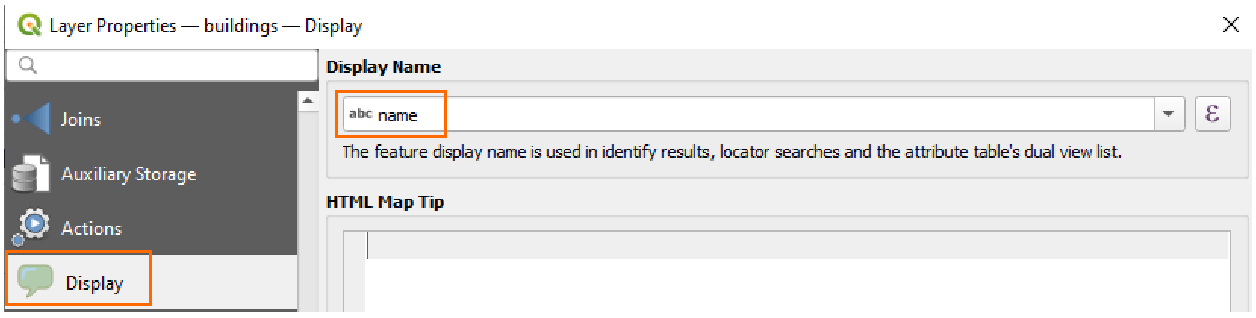

Organize Table columns

- ✓ osm_id
- ✓ city
- ✓ street
- ✓ number
- ✓ search_text
- ✓ code
- ✓ type

Figure 159 – 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 160 with the example.


Layer Properties — buildings — Display

Display Name

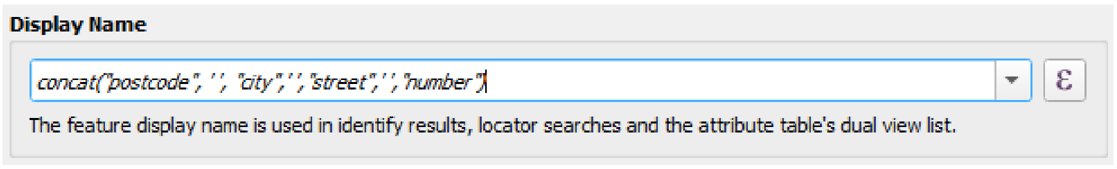
abc name

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

HTML Map Tip

Figure 160 – 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 161.


Display Name

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 161 – Display by multiple fields

Note: only following functions are supported: "sin", "cos", "tan", "atan", "abs", "asin", "acos", "log", "log10", "cailing", "floor", "round", "exp", "sqrt", "ltrim", "rtrim", "substr", "substring", "concat", "lower", "upper", "pow", "power", "andbits", "len", "length", "coalesce", "mod", "scale_exp", "scale_linear", "to_string", "tostring", "var", "min", "max", "now", "interval", "date_part"

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.

Additional parameters for fine tuning of the geocoding

`elitegis_geocode_search_fields`

- List of fields for search. By default the free text search is done by the field specified in Display Field. If the expression is specified, free text will not work. In this case you need to specify the fields by which the search will be performed using this property (i.e. fields that were specified by creation of the free text index and in the same order).

`elitegis_geocode_replacement_exact` и `elitegis_geocode_replacement_words`

- The string from the set of replacement pair "whattoreplace=bywhattoreplace;", by default nothing is changed
- for pre-replacement of symbols or words in the search string before invoking the search in the database, it is needed to:
  - ♦ remove specific symbols (points and commas), to turn slash for data and other "exact" replacements of a symbol or set of symbols, wherever they occur in the source string
  - ♦ replace abbreviation with the full name, so that the request correspond to the data and so that full text search could find anything

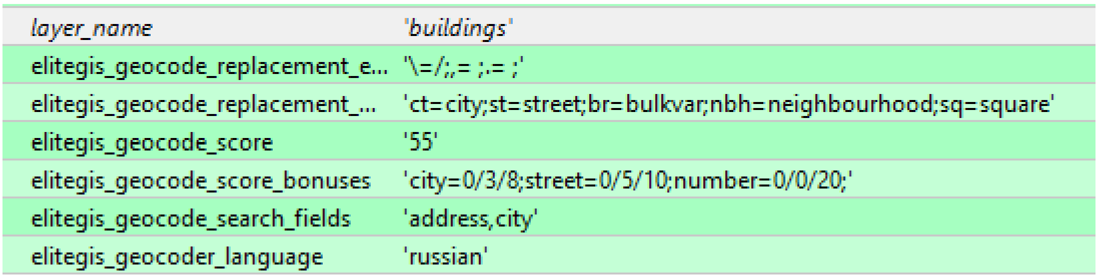
`elitegis_geocode_score`

- ♦ minimum score that will be used by search via the full text index; integer, by default 100

`elitegis_geocode_score_bonuses`-

- ♦ the option to increase the score by the comparison of values in the fields of the found result with the requested string (case insensitive), the string from the set of pairs "FIELDNAME=BONUS_CONTAINS_A_WORD/BONUS_CONTAINS_A_WORD/BONUS_CONTAINS_A_WORD/BONUS_CONTAINS_A_WORD;"

These properties are specified on the Variables tab, see Figure 160.



layer_name	*'buildings'*
elitegis_geocode_replacement_e...	'\=/;,= ;.= ;'
elitegis_geocode_replacement_...	'ct=city;st=street;br=bulkvar;nbh=neighbourhood;sq=square'
elitegis_geocode_score	'55'
elitegis_geocode_score_bonuses	'city=0/3/8;street=0/5/10;number=0/0/20;'
elitegis_geocode_search_fields	'address,city'
elitegis_geocoder_language	'russian'

Figure 162 – Setting the field for search

Table 19 – Example of values for geocoder properties

Parameter	Example of value
elitegis_geocode_replacement_exact	\=/;,= ;.= ;
elitegis_geocode_replacement_words	ct=city;st=street;br=bulvar;nbh=neighbourhood;sq=square;
elitegis_geocode_score	65
elitegis_geocode_score_bonuses	street_type=0/0/5;street_name=0/5/10;house_number=0/0/20;
elitegis_geocode_search_fields	district_name,full_address

Now save the created QGS project, see Figure 163.

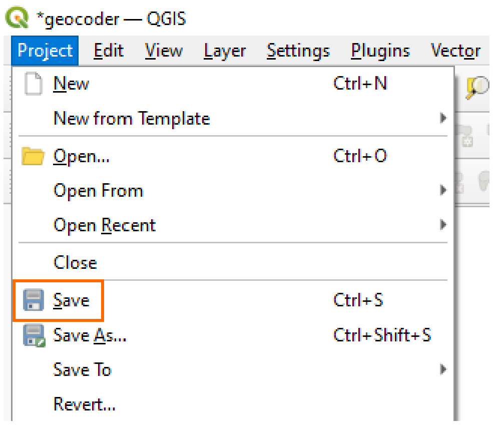


Figure 163 – Saving QGS project

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

7.3. Publishing the geocoding service in CoGIS Server

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

For publishing geocoding service created based on the map project (see above), open **CoGIS Server Manager** and download QGS project, see Figure 164.

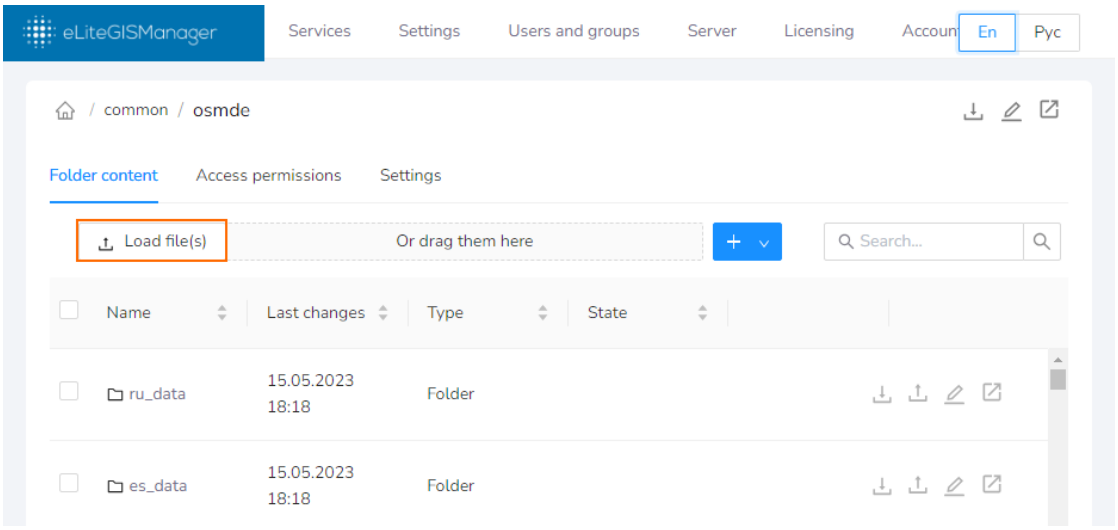
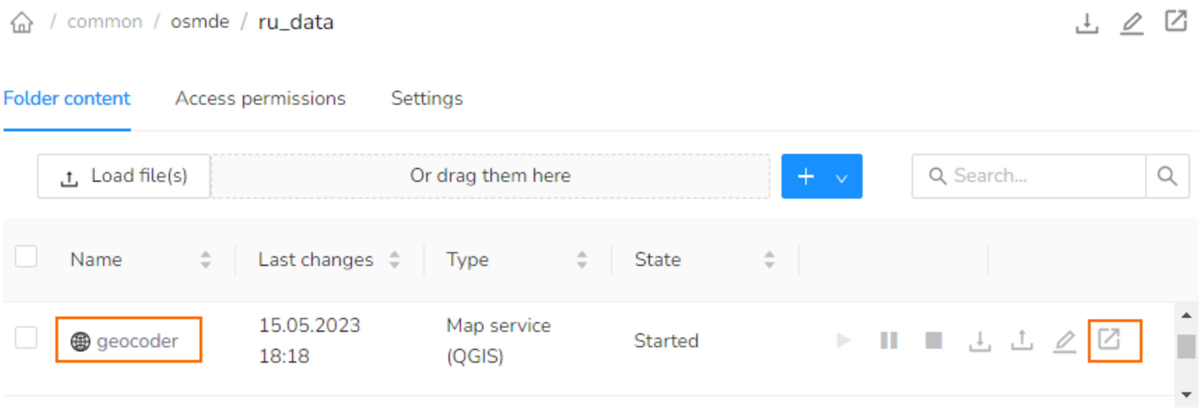


Figure 164 – Downloading QGS project in CoGIS Server

After the project is downloaded, the map service is automatically published. Check its working capability as shown on Figure 165.


/ common / osmde / ru_data

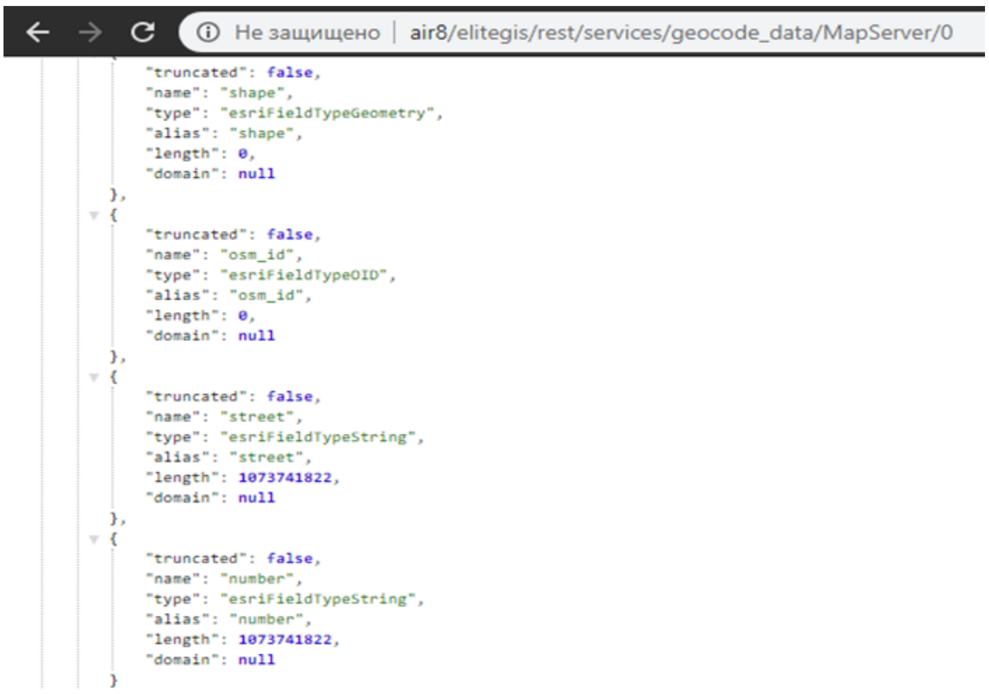
Folder content | Access permissions | Settings

Load file(s) | Or drag them here | + | Search...

Name	Last changes	Type	State
geocoder	15.05.2023 18:18	Map service (QGIS)	Started

Figure 165 – Checking the geocoding service working capability

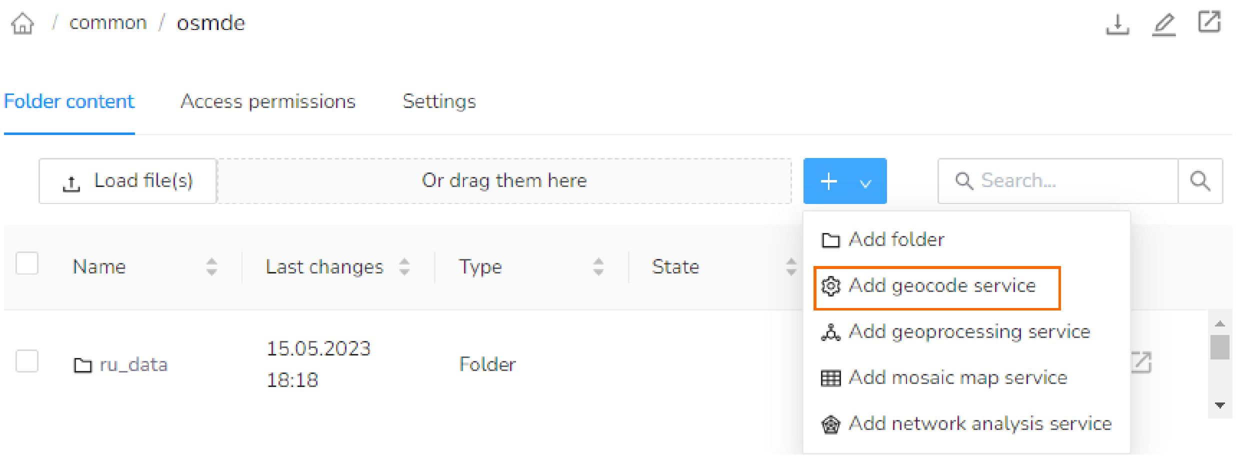
The order of fields in the service declaration should be the same as for the freetext index, see Figure 166.


Не защищено | air8/elitegis/rest/services/geocode_data/MapServer/0

```
      "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 166 – Checking the fields order in the service declaration

After checking completion, the geocoding service can be published, see Figure 167.


/ common / osmde

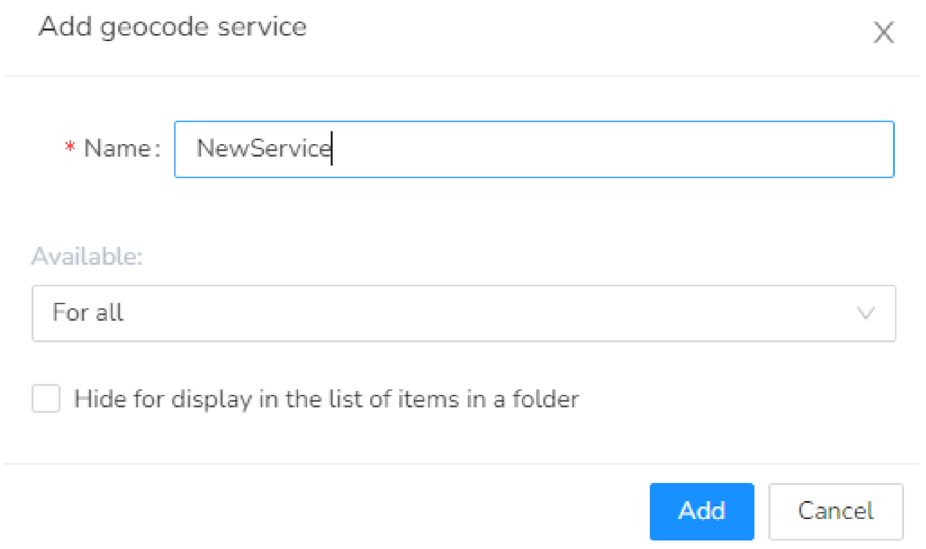
Folder content | Access permissions | Settings

Load file(s) | Or drag them here | + | Search...

Name	Last changes	Type	State
ru_data	15.05.2023 18:18	Folder	

Figure 167 – Adding the geocoding service (1)

In the appeared window specify the service name and press Add button, see Figure 168.


Add geocode service

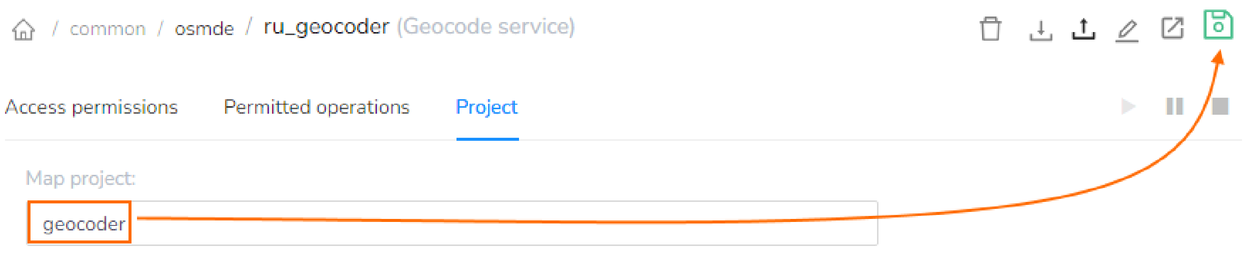
* Name: NewService

Available: For all

Hide for display in the list of items in a folder

Figure 168 – 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 169.



Map project: geocoder

Figure 169 – Setting the geocoding service

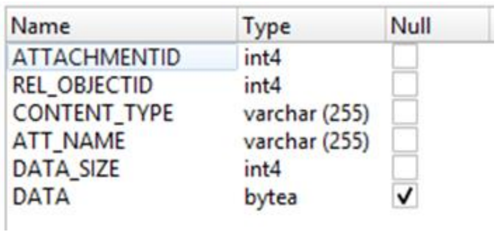
Geocoding service is ready for work.

8. Attachments

CoGIS Server 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 170.



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

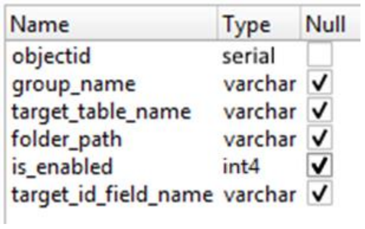
Figure 170 – Structure of table for storage of attachments in the database

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 171.



Name	Type	Null
objectid	serial	☐
group_name	varchar	✓
target_table_name	varchar	✓
folder_path	varchar	✓
is_enabled	int4	✓
target_id_field_name	varchar	✓

Figure 171 – Structure of table for storage of attachments as files on the disk

The table:

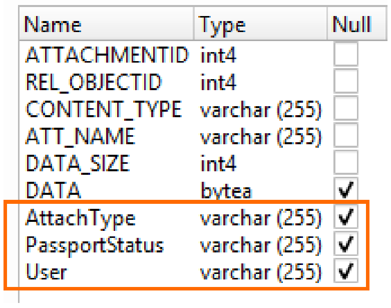
- *group_name* – the name of table record for orientation (can be not unique);
- *target_table_name* – the short of full name or template of the feature class name;
- *folder_path* – the path to the root folder where to the files should be saved;
- *is_enabled* – the property indicating whether the setting of this record should be used (0 - no, 1 - yes);
- *target_id_field_name* – which field should be used for identification in the names of folders for storing attachment; if null, then the oid field is used.

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,
  target_id_field_name character varying,
  CONSTRAINT elitegis_attachment_groups_pkey PRIMARY KEY (objectid)
);
```

8.3.Attachment attributes

For attachments it is possible to add additional attributes, for example, Attachment type, Attachment author, etc. These attributes are added as additional fields to the attachments table, see Figure 170.



Name	Type	Null
ATTACHMENTID	int4	☐
REL_OBJECTID	int4	☐
CONTENT_TYPE	varchar (255)	☐
ATT_NAME	varchar (255)	☐
DATA_SIZE	int4	☐
DATA	bytea	✓
AttachType	varchar (255)	✓
PassportStatus	varchar (255)	✓
User	varchar (255)	✓

Figure 172 – Example of additional fields in the attachments table

In order that the additional attributes for attachments could be available in the service, it is required to add the attachment table to the QGIS project, see Figure 171.

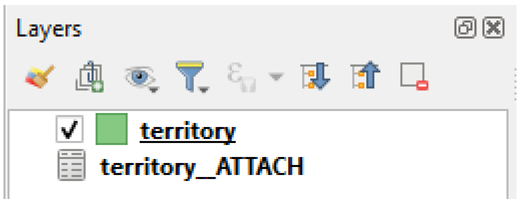
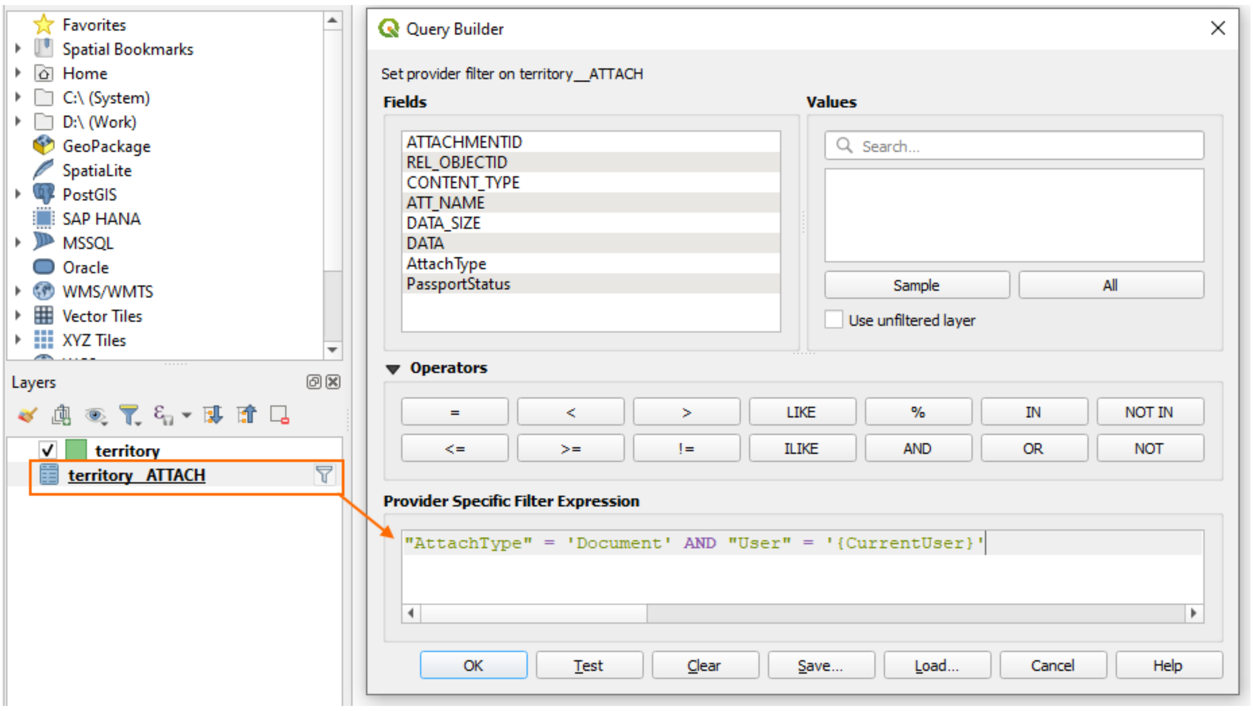


Figure 173 – Example of adding attachment table to QGS project

8.4. Attachments filtration

CoGIS Server supports the ability to filter attachments at the service level by primary or secondary attributes. The filter should be specified for the attachments table added to the QGIS project.
An example of adding filter for attachments is shown on Figure 172.



```
"AttachType" = 'Document' AND "User" = '{CurrentUser}'
```

Figure 174 – Example of attachments filtration

Besides, some macros for expressions from section 5.1.4 are available:

- Filtration by user, macros *{CurrentUser}*;
- Filtration by group of users, macros *{CurrentGroup.<group name>}*.

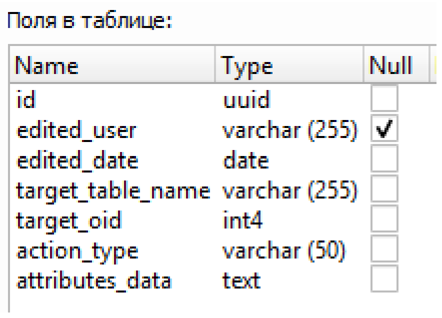
8.5. Returning attachments table for the layer

There is the possibility to prohibit returning attachments for the layer if they are configured for it in the database.
Property *elitegis_attachments_enabled: true/false* (by default *true*).
If *false* is set, then attachments for the layer will be disabled.

9. Edits history

CoGIS Server 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 the work of the edits history setting, the corresponding SOE rule (see details in the **CoGIS Portal - Creating map applications** 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 175:



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 175 – 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)
);
```

10. Tiles auto update

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

Such an image is divided into blocks/squares called tiles.

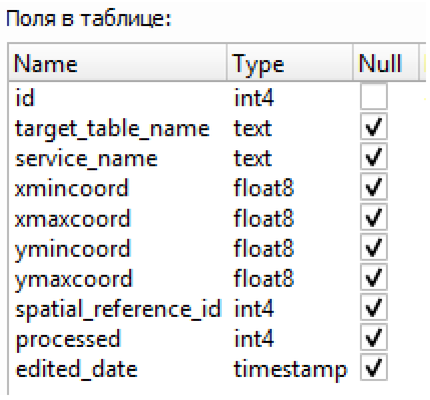
After generation, the tiles are stored in the tile cache, and on subsequent requests, the tiles are not re-generated, but taken from the cache. Thus, if map features have been changed, then these changes will not be reflected on previously generated tiles.

To avoid such situations, the map service in **CoGIS Server** can be configured to update those tiles that contain changed features. Besides the corresponding setting of service in **CoGIS Server Manager** (see **CoGIS Server - Publishing GIS services** 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 176 – 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)
);
```

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();
```