Working with MultiSelect queries

CONTENTS

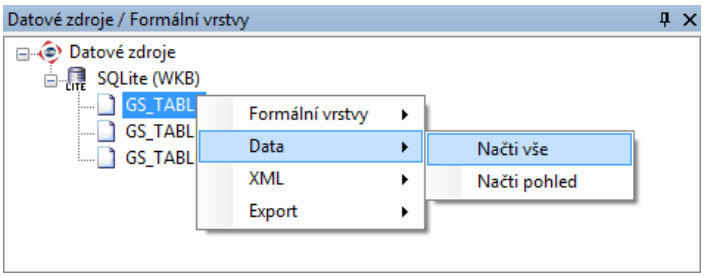
1 AIM OF THE EXAMPLE....2
2 WORKING WITH EXAMPLE 2
3 DIALOG BOX SAMPLE....3
4 A BRIEF DESCRIPTION OF THE EXAMPLE IN MARUSHKADESIGN 6

1 Aim of the Example

In this example, we will demonstrate work with MultiSelect and finding information about properties of the elements included in this selection. This example was created in version 4.0.1.0, so it does not have to be compatible with older versions.

2 Working with Example

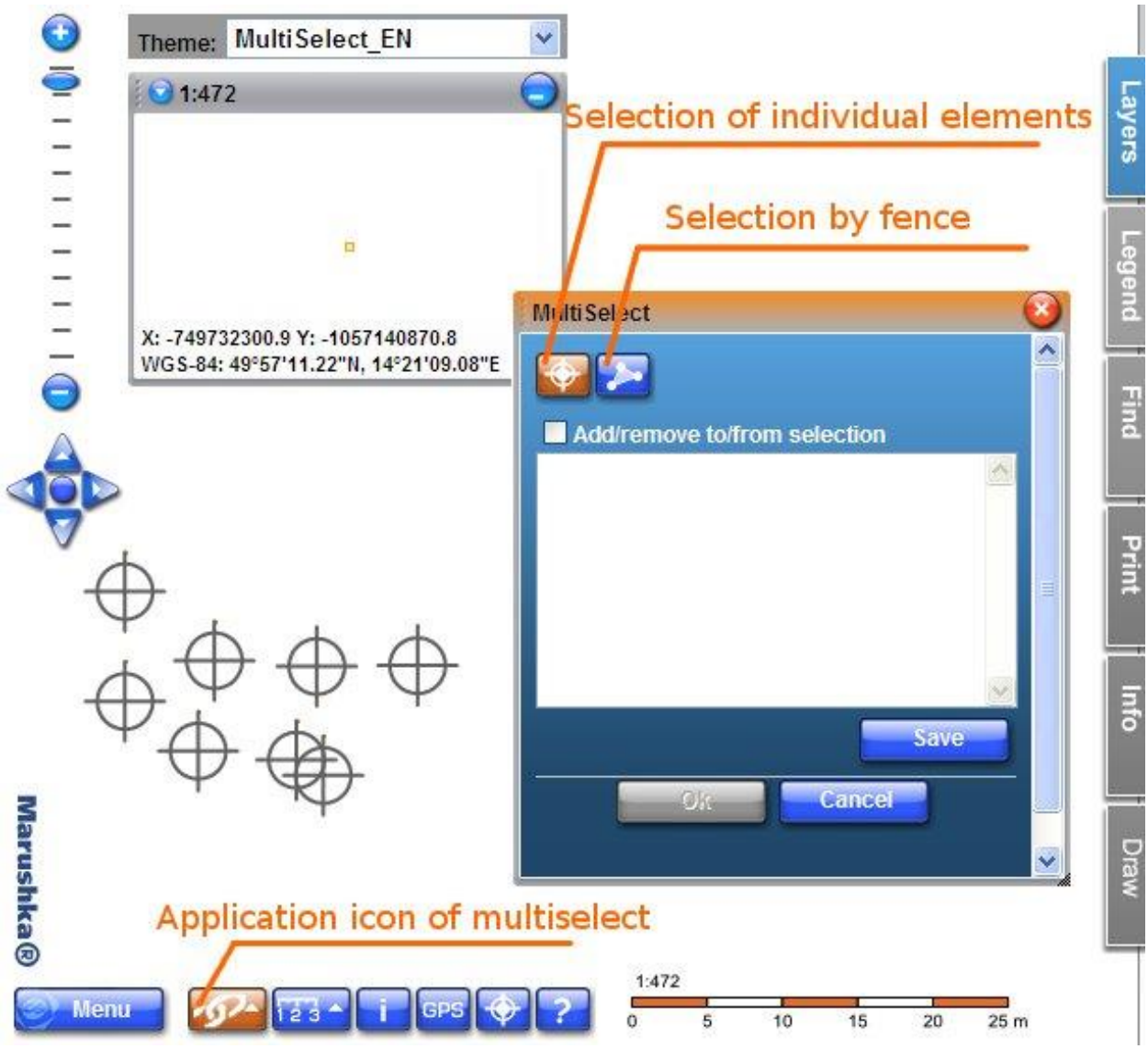
- Unzip the content of file **MultiSelect_EN.zip** into folder **c:\MarushkaExamples**. The target folder must be respected due to interconnection of paths with the project. In the case of placing the files in different folder, it would not be possible to work with an example.
- Open the project **MultiSelect_EN.xml** in the MarushkaDesign environment.
- Select the form layer GS_TABLE, in context menu choose Data – Load all:



- In map window choose "Fit all":

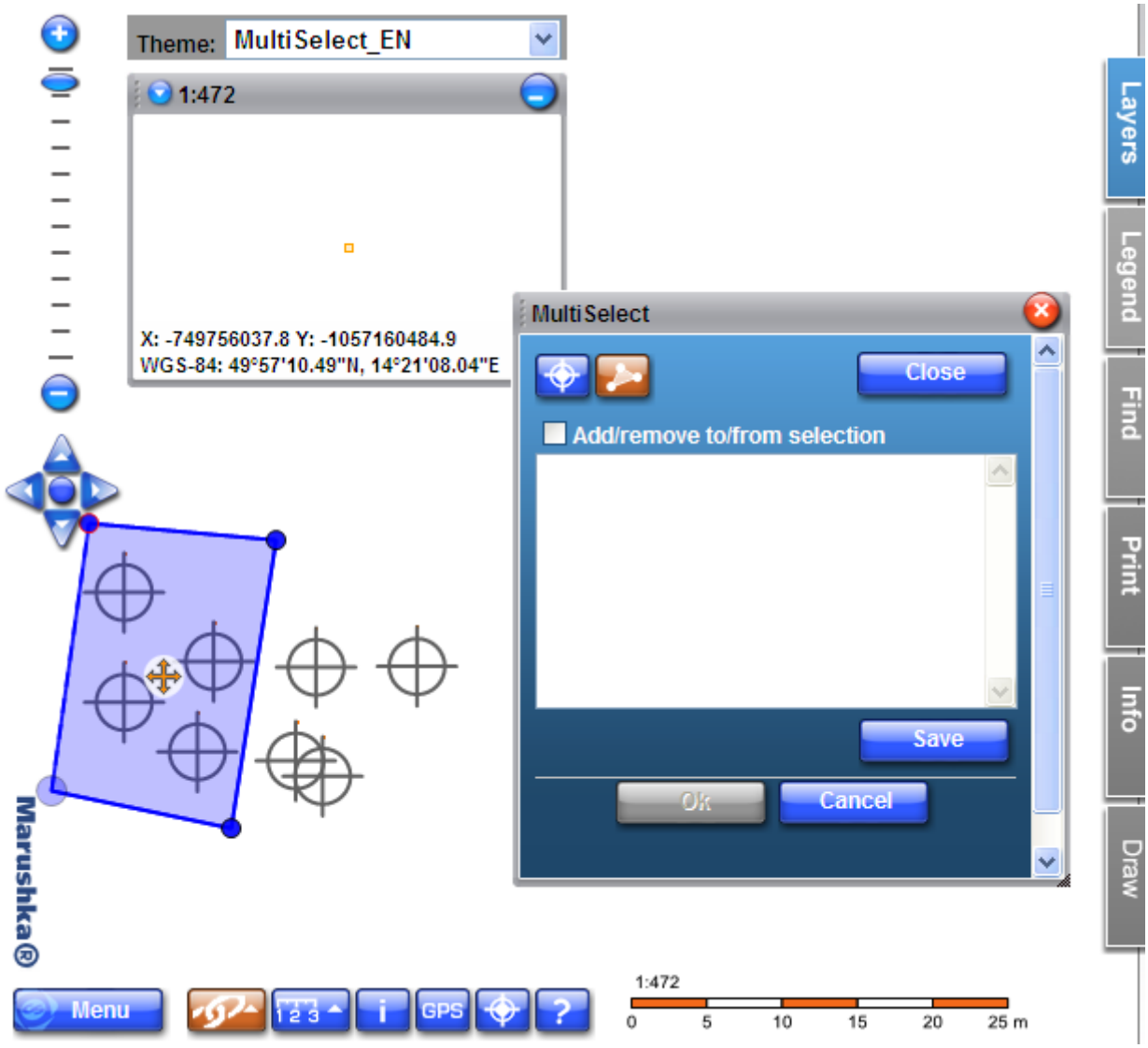
- Launch the local web server:

3 Dialog Box Sample

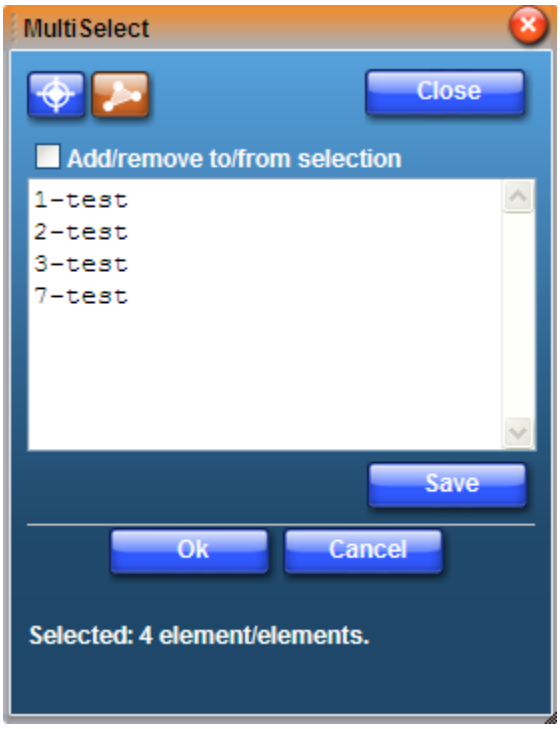


To work with query type Multiselect, the most important feature is an application icon, which may be represented by any image. Click on this icon will start the multiselect function. Individual elements can be selected individually or by fence. In case, when the option "*Add/ remove from/to the selection*" will not be checked during the selection, then before each selection the selection set will be cleared and you will have to choose element (or group of elements) again.

In the following picture is an example of selection by fence, when we gradually enter individual vertices of the selection polygon:



The selection polygon is closed by the Close button. The information about the selected elements will display, in our case the description has this form: `ID-notice`. We can save information about an element into a text file using the Save button.



At the moment we have defined the whole selection set, click on OK, which will start the information query defined for the MultiSelect query. The result of our query will have the following form:

4 A Brief Description of the Example in MarushkaDesign

The test example contains a database in SQLite with one publish layer. There are three form layers in the SQL data source.

Layer **GS_TABLE** is used for drawing map composition.

For MultiSelect are used these two layers: Layer **GS_TABLE highlight** is used to highlight the selected elements, i.e. it defines the symbology of selected elements and will be subsequently drawn into the map compositions with the highest *LoadOrder* (above all layers). **GS_TABLE select** is only used to select elements, i.e. it defines the data that is performed spatial analysis for and selects elements contained in the selection polygon (MultiSelect by fence) or elements that are near the point of click (selection by point) usually represents the raw data without symbology.

MultiSelect work as follows:

a) Conducts spatial analysis (by polygon or point) above the *GS_TABLE select* layer. The output is a list of IDs of elements that are placed inside the selection polygon, or are located near the point of click.

b) A list from the previous step is used for rendering of selected elements in a way that in the *GS_TABLE highlight* form layer is automatically added condition *ID in* (list of IDs from the previous step).

Finally, a test example consists of a pair of queries. The first one is the **Information** query that serves as a source of information of individual elements; on input this query receives a list of selected elements. This list is a result of the query "Multiselect".

The only significant difference compared to the standard information query is, that in SQL phrase is a set of elements whose type must be `~(idlist)ID~` instead of standard `~(long)ID~` in common information queries. In this example, there is the definition of property *SqlStmtTemplate* in this form:

```
SELECT notice FROM GS_TABLE WHERE ID in (~(idlist)ID~)
```

The main query is the **MultiSelect** query, which is created in the Query library by right click on the appropriate data store, *New* – *Utility* – *MultiSelect*. There is an important property "IsApplication", which must be set to value *"True"* to display the query in the publication under the icon "External Application".

Essential are mainly layers for the highlight – property ***HightlightFormLayer*** and the layer above which is the spatial analysis performed – property ***SelectFormLayer*** and subsequently interconnection with the information query – property ***Query***.

Another property that MultiSelect query can include is ***ShowSelectedLabels***. This allows us to see a list of already selected elements in the text form in the component of MultiSelect. By default for the label is used ID value of the element. This value can be changed by the definition of property ***LabelAttributeName***. This property defines the name of the attribute of the selected element, whose value is then used as a label (this attribute/ column must be defined in property *DBColumnsToClient* of form layer *GS_TABLE select*. By this will be ensured that while loading the element, also the relevant attribute will be loaded).

Furthermore, if we set the value of ***ShowSelectedLabelsSaveButton*** property to "*True*", then in the MultiSelect window will appear the button for saving a list of currently selected elements into the text a file.