Using Dynamic Environment Parameters SET_ENV_ in MarushkaDesign

GEOVAP

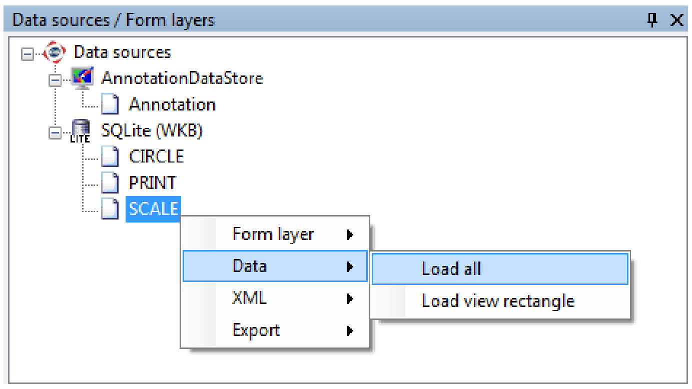
CONTENTS

1 Aim of the Example

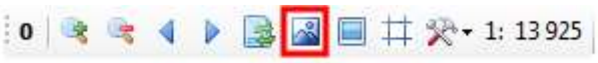
In this example, we will demonstrate work with dynamic environment parameters in MarushkaDesign. This example was created in version 4.0.1.0, so it does not have to be compatible with older versions.

2 Working with Example

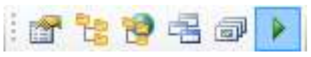
- Unzip the **SET_ENV_EN.zip** into **c:\MarushkaExamples\** folder. The target folder must be respected due to interconnection of paths with the project. In case of placing the files in the different folder, it would not be possible to work with an example.
- Open the **SET_ENV_EN.xml** in MarushkaDesign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



- In map window choose “Fit all”:

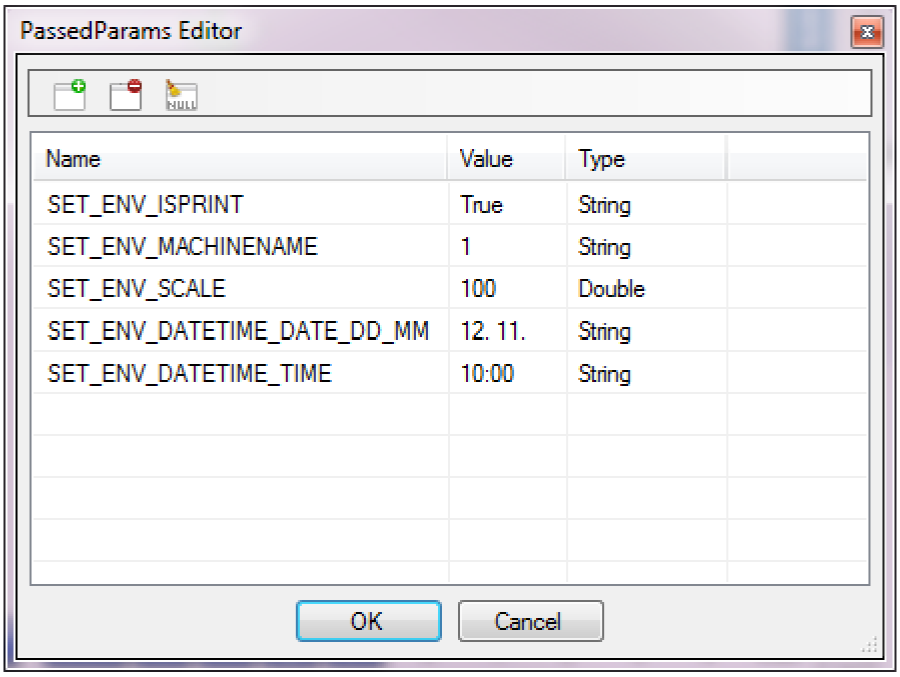


- Launch the local web server:



3 Dialog Box Sample

Fig 1: Sample of parameters definition SET_ENV_ in PassedParams editor


PassedParams Editor

Name	Value	Type
SET_ENV_ISPRINT	True	String
SET_ENV_MACHINENAME	1	String
SET_ENV_SCALE	100	Double
SET_ENV_DATETIME_DATE_DD_MM	12. 11.	String
SET_ENV_DATETIME_TIME	10:00	String

Fig 2: Sample display of form layer result in the local web server environment

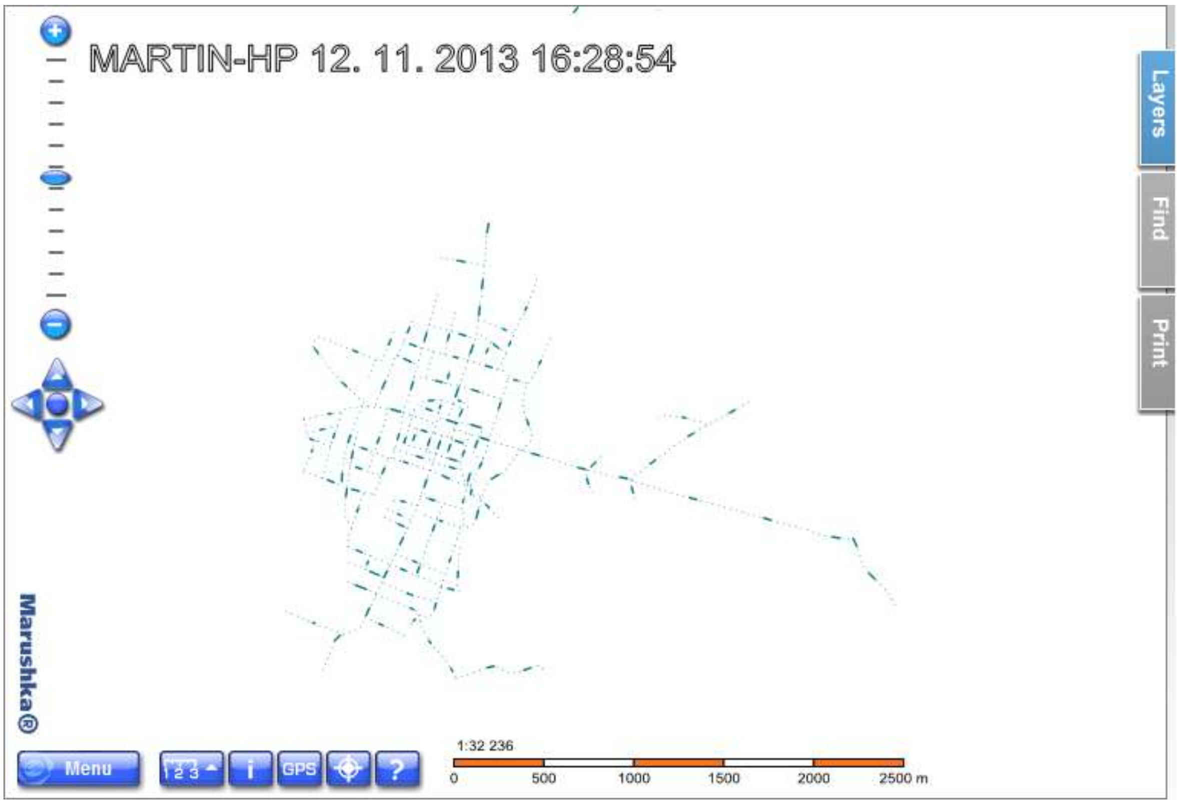
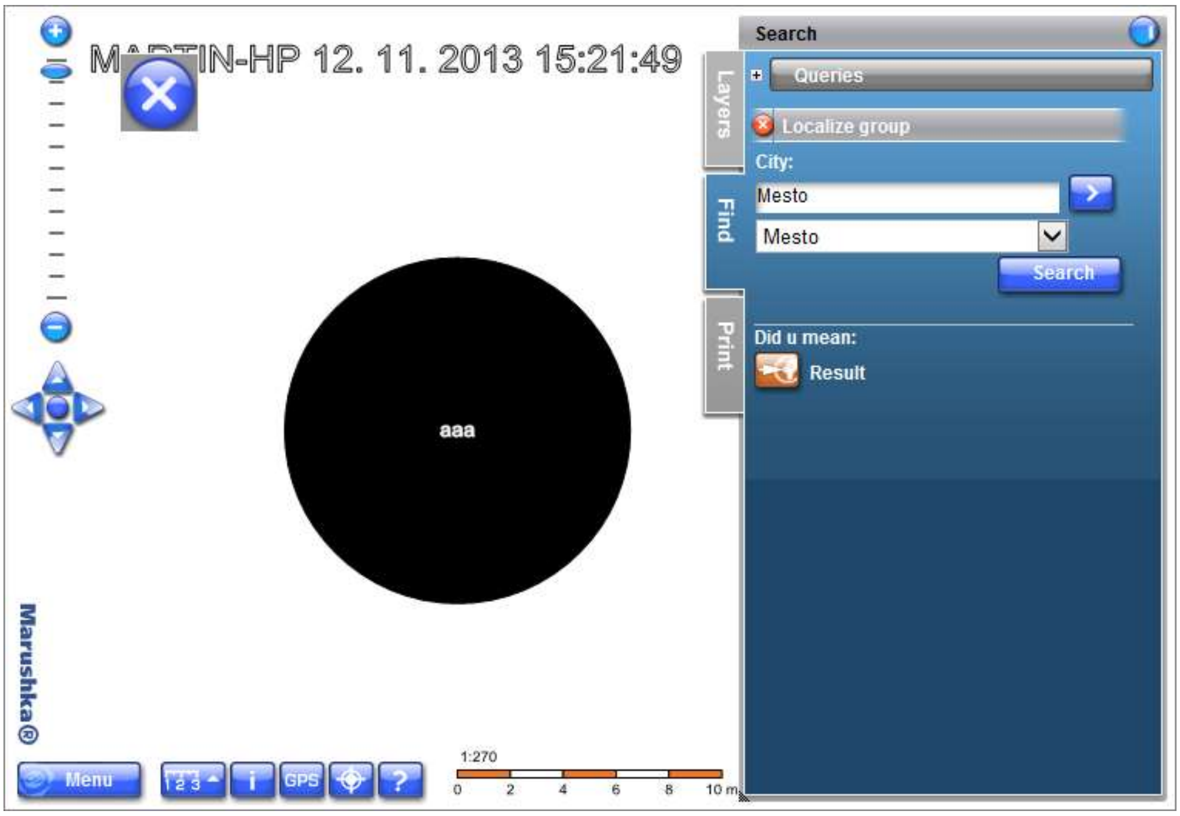

MARTIN-HP 12. 11. 2013 16:28:54

Fig 3: Example of localization query result *Localize Group*



4 A Brief Description of the Example in MarushkaDesign

Dynamic parameters SET_ENV_

Individual parameters *SET_ENV_* are environment parameters, which are used for substitution into individual database queries. When using any of these parameters, it is also necessary to define such parameters into *PassedParams* of the project and define the default values for them. Without this definition it could happen, that the server would not know the value of the parameter *SET_ENV_* and the result could end up in an error condition.

The values of these parameters cannot be changed by the user from the outside; they are dynamically substituted by the server (their value changes over time) in the event that they occur in a database query (e.g. calling the print job for `SET_ENV_ISPRINT`).

Here is a list of supported *SET_ENV_* parameters in the current version of MarushkaDesign (4.0.1.0):

SET_ENV_SCALE: Parameter used to define the scale range of the displayed layer, format: *Double*

SET_ENV_ISPRINT: Parameter that allows different behavior of publication in the case when the print job is invoked, format: *Boolean*

SET_ENV_MACHINENAME: Parameter that contains the user computer name, format: *String*

SET_ENV_THEMEID: This parameter contains the ID of the current theme, format: *Int*

SET_ENV_USER: Parameter that can interact with the user name of authenticated user (if authorized access to the database data store), format: *String*

SET_ENV_DATETIME_DATE_DD_MM: Parameter containing the current date in format: *DayDay MonthMonth Year*, format: *String*

SET_ENV_DATETIME_DATE_MM_DD: Parameter containing the current date in: *MonthMonth DayDay Year*, format: *String*

SET_ENV_DATETIME_TIME: Parameter containing the current time in format: *HourHour MinuteMinute SecondSecond,* format: *String*

This example contains 3 form layers and 1 annotation layer. Specific use of dynamic parameters *SET_ENV_* is shown in the following sections:

1. Parameter **SET_ENV_ISPRINT**. Form layer *PRINT* renders the street lines of the three municipalities with descriptions. *In DBColumnsToClient* this form layer has set the following string:

   ```
   ~(string)SET_ENV_ISPRINT~,CASE WHEN ~(string)SET_ENV_ISPRINT~='True'
   THEN '255 255 128 0' ELSE '255 125 125 0' END SET_PARS_RGBCOLOR
   ```

   This means that after invoking a print job on the tab "*Print*" and clicking on the button "*Next*", will all the elements from this form layer recolor to orange color. It would be possible to set, so that in print will be displayed only the line elements from the database or vice versa, that in print will be added a new element (such as watermark). This parameter has set as the main type *Bool*, but it can be used successfully even if it is the *String* type (this was used in the example).
   If the default value of the parameter `SET_ENV_ISPRINT` was not defined in *PassedParams* of the project, it would not be possible to perform operations with form layer *PRINT*, such as *Data – Load all* or *Data – Load view rectangle*; because the server would not know what value to substitute instead of a parameter and a result would have ended by syntax error.

2. Parameter **SET_ENV_MACHINENAME** is substituted by the user name of the computer. Here, this parameter is displayed using annotation layer *Annotation*. The actual annotation is then displayed in the upper left corner of the map window together with layers *SCALE* and *PRINT*, and specifically in this example it displays the text MARTIN-HP. The way how to use this parameter in the annotation layer is somewhat unusual, because it is not put into the item properties *DBColumnsToClient*, but into the item *Text*, into which was entered a string `~SET_ENV_MACHINENAME~`. In this MarushkaDesign version, it is possible to use this parameter in annotation layer only untyped, because it would not substitute the user's computer name, but only the entered string would be displayed.

   You can try yourself, if you are receiving the correct machine name. In Windows 7, click start *Start* / right click on *Computer* / *Properties* / and here is your *Computer name*, this name should match the name displayed by the annotation layer.

3. Parameter **SET_ENV_SCALE** is substituted by the value of the current map scale. Here it is specifically used for form layer *SCALE*, which has in *DBColumnsToClient* set using *CASE* dynamical change of color of displayed elements of the street network at different scales. Specifically, here is an ensemble of 5 categories, each category has set a different scale range and display color. The whole SQL query, using which was this layer behavior set, is defined in *DBColumnsToClient* of form layer *SCALE*.

   The whole query looks as follows:

   ```
   ~(double)SET_ENV_SCALE~, CASE WHEN ~(double)SET_ENV_SCALE~<5000 THEN
   '255 255 0 0' WHEN ~(double)SET_ENV_SCALE~<10000 THEN '255 0 0 255'
   WHEN ~(double)SET_ENV_SCALE~<25000 THEN '255 255 0 255' WHEN
   ~(double)SET_ENV_SCALE~<50000 THEN '255 0 125 125' ELSE '255 0 255
   0' END SET_PARS_RGBCOLOR
   ```

   You can try yourself the functionality of color change using zoom-in / zoom-out using the mouse wheel, when layer *SCALE* is checked in the local web server.

4. Again using parameter **SET_ENV_SCALE** it is possible to create a more complex localization query *Locate group*. If the query did not include this parameter, the query would have been created using two form layers.

   This localization query has set, that query result renders black circle in the real size and into its reference point it renders text 'aaa', in a fixed pixel size of 10 pixels. But this text is rendered only up to scale 1 : 2 500, because at a higher scale, the text is larger than the cell, resp. the cell is smaller than the text.

   In this query, the most important construction is `UNION ALL`, which combines both parts of the query. However, it does not discard duplicates, so when using it, you must remember that associated queries should have the same columns. Using the pseudo columns was set reference point for displaying text, fill color, text, and text height, switching on text height in pixels and cell name that is displayed in the localization result.

   In the case that the default value was not specified in *PassedParams* of the project, the query would not be functional. Before evaluating the query, at the time when possible occurrences are searched, the current scale is not available. And therefore, instead of the current map scale, the default value from *PassedParams* is substituted, which is then compared with condition in *WhereClause*. When evaluating the result, the condition is already compared to the current value of the scale.

   The whole query looks as follows:

   ```
   select xmin-20 XMIN, ymin-20 YMIN, xmax+20 XMAX, ymax+20 YMAX, id
   ID, '0 0 7' SET_PARS_POINT_FROM_CORG,'' SET_PARS_RGBFCOLOR, geom
   GEOM, null SET_PARS_TEXT, 'Bunka' SET_PARS_CELLNAME, null
   SET_PARS_HEIGHT,'FALSE' SET_PARS_PIXELSIZE FROM GS_TABLE WHERE City
   like '~1~'
   ```

```
UNION ALL

select xmin-20 XMIN, ymin-20 YMIN, xmax+20 XMAX, ymax+20 YMAX, id
ID, '0 0 7' SET_PARS_POINT_FROM_CORG, '255 255 255 255'
SET_PARS_RGBFCOLOR, geom GEOM, 'aaa' SET_PARS_TEXT, null
SET_PARS_CELLNAME,10 SET_PARS_HEIGHT,'TRUE' SET_PARS_PIXELSIZE FROM

GS_TABLE WHERE City like '~1~' AND ~SET_ENV_SCALE~<2500
```

5. Since Marushka version 3.0.16.0 it is possible to use three new types of parameters SET_ENV_. These include **SET_ENV_DATETIME_DATE_DD_MM**, **SET_ENV_DATETIME_DATE_MM_DD** and **SET_ENV_DATETIME_TIME**.

   There is also a new possibility of combining SET_ENV_ parameters, when it is possible to combine them in either database queries or for example in property *Text* of annotation layer. As the demonstration is given an annotation layer, which have in property *Text* defined text string:

   ```
   ~SET_ENV_MACHINENAME~ ~SET_ENV_DATETIME_DATE_DD_MM~
   ~SET_ENV_DATETIME_TIME~
   ```

   and it therefore displays in the left upper corner current computer username, date in format: *DayDay MonthMonth* and time in format: *HourHour MinuteMinute SecondSecond*. Yet you need to set the property *Type* to *Text* to display substituted parameters.