Tile System in Marushka Design

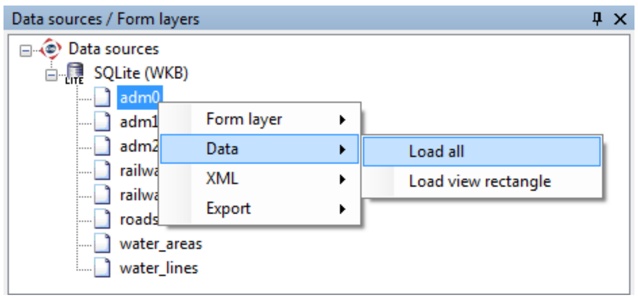
CONTENTS

1 Aim of the Example

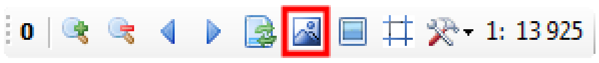
In this example you will be shown how to set up the Tile system in MarushkaDesign, so that the map image is cut into tiles, which are then displayed in Marushka. This example was created in version 4.0.2.38, so it does not have to be compatible with older versions.

2 Working with Example

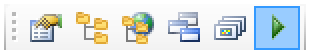
- Unzip the **Tile_EN.zip** into **c:\MarushkaExamples\** folder. The target folder must be respected due to interconnection of paths with the project. In the case of placing the files in the different folder, it would not be possible to work with an example.
- Open the **Tile_EN.xml** in MarushkaDesign environment.
- Select form layer **adm0**, in the context menu choose *Data – Load all*:



- In map window choose "Fit all":

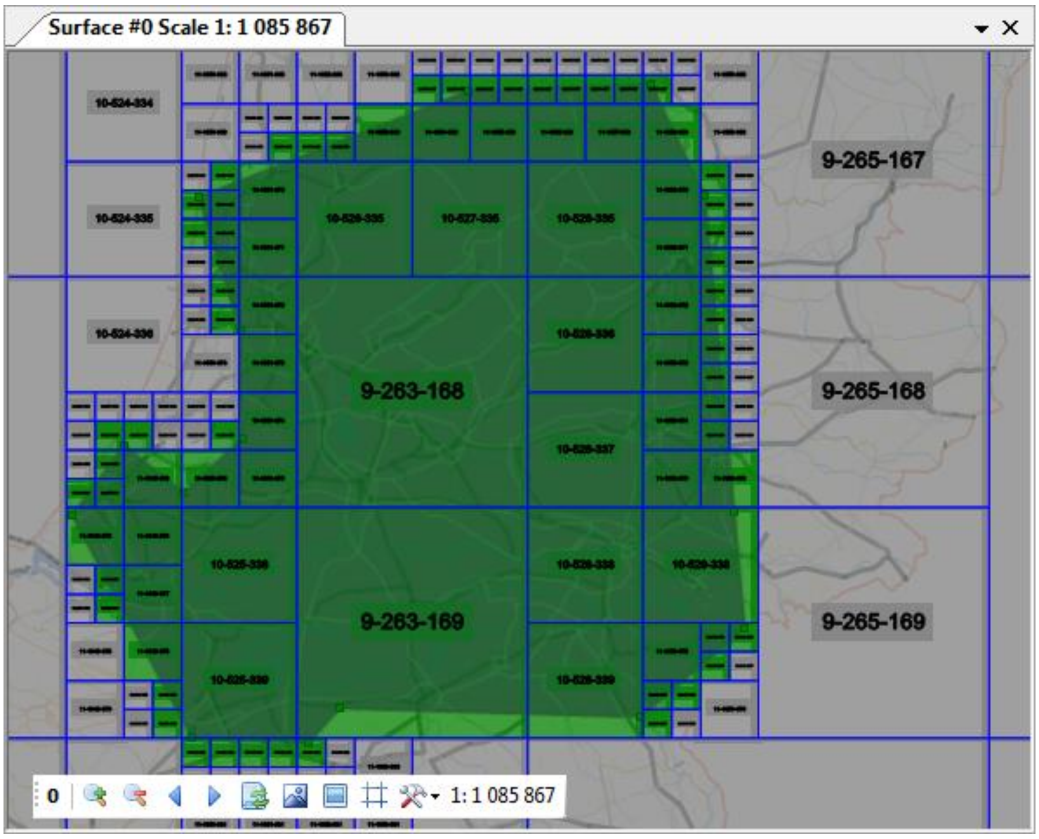


- Launch the local web server:

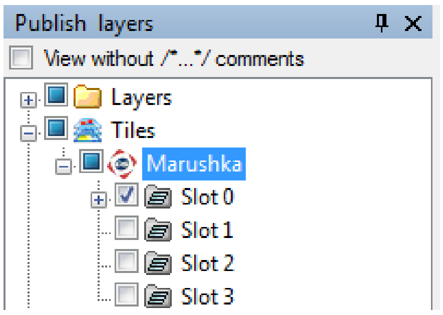


GEOVAP

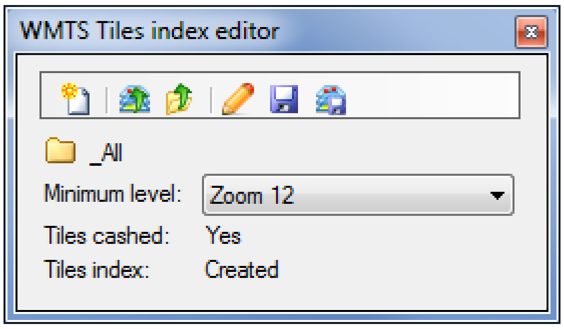
3 Dialog Box Sample



Img 1: Tile index sample in the map window



Img 2: Publish layers dialogue box for tile layer settings


WMTS Tiles index editor

_All

Minimum level: Zoom 12

Tiles cashed: Yes

Tiles index: Created

Img 3: WMTS Tiles index editor

4 A Brief Description of the Example in Marushka Design

This test example contains a database in SQLite with eight form layers. These layers display administrative borders, railway lines, roads, water lines, and water areas in Nederland. It also contain a popular publish tile layer, which groups all the listed form layers in one tile layer, called ***_All***.

Principles and Properties of Tile Layers

Principle of tile layers consists in the fact of loading whole elements in the map window. The map image is then cut into tiles (standardized images), which are based on the spatial query subsequently returned. It is therefore not necessary to create a map image so that whole elements are loaded, but the map image is folded from these standardized images, which were created from these cropped elements.

Like e. g. in Google maps tiles have strictly defined zoom levels, therefore they does not have continuous scale. In each of such defined levels is the map image cut into individual tiles. The first level contain only one tile, the second is created so that the tile from the first level is cut into 4 same pieces. Similarly, the third level will contain four times more tiles than the second level (i.e. 16), because each of tiles is again cut into 4 pieces.

Among the indisputable advantages of tiles is quicker load of map elements, because into the map window are not loaded whole elements, but only the pre generated image tiles are loaded as images in a specific format. From those is then folded the whole map window image. The advantage over the Google maps is that tiles can be created in any projection.

Due to the discontinuous scale, it is not possible to set a specific scale in the project. But always is offered the next available from zoom scales. However, in print it is possible to set any specific scale, since printing is generated from the source data and not from the generated tiles.

Publish Layer Settings

It is possible to set a publish layer in category *6. OGC Web Map Tile Service,* item *AllowWMTS* to value *“True”*, which causes, that the given publish layer will be cut into tiles that will be returned by webserver. In this project, this item is set to value “*True*” for publish layer *_All*, which display all the form layers from this data store. To ensure that the tiles were generated and stored on the server, it is necessary that this publish layer should have also set item *WmtsRootFolder*, which sets the folder in which are saved the generated tiles.

It is also necessary in publish layer dialog box to right mouse click on item *Tiles* and select *Marushka tile engine*. Next the user have to right click on one of slots, chooses the option *Add new Tile Layer* and then select some publish layer, which has set attribute *AllowWMTS* to value *“True”*.

These layers can be then adjusted similarly to Google map layers using various attributes that influence their behavior: if these will be popular, visible, default checked, and other setup items typical for publish layers in general.

Possibilities of Saving into Cache

The user can specify whether the tile layer from the publish layer will be stored on disk or whether these will be discarded right after generating them. It is set using parameter *WmtsTilesClientCache*. If this parameter is set to value ‘*True*’ and the *WmtsRootFolder* value would not be set, the generated tiles will be saved just into operating memory. However, if simultaneously is set the *WMTSRootFolder* value, the generated tiles are stored on the disc. Reloading the tiles stored on the disc will be several times faster than generating new tiles.

WmtsTilesCashedMinLevel – Sets the minimum level up to which the tiles are stored on the disc, all the other levels will be then generated dynamically. This saves disc space in the case of large number of elements in lower scales.

WmtsTilesClientCache – Setting of caching on the client side; allows storing the generated tiles on the disc. If another request on the same tile will occur, it will proceed much faster than it takes to generate the tile. This feature is suitable especially for web applications.

WmtsTilesClientCacheExpireTime – It is the time in minutes, during which the tile is kept on the server. After this period will the validity of tile expire and the client would make a new request on the server.

Indexes

For tile layers, it is possible to create an index that determines in which area are to be generated tiles. In other areas, tiles will not be generated. This measure serves to not unnecessarily generate blank tiles, which would decrease the computer performance.

For work with indexes was created *WMTS Tiles index editor*, see Img 3.

- Creates new tile index from selected polygon
- Loads symbol from root file with tiles
- Loads index from directory
- Displays index structure in the map window
- Saves index
- Saves index into the root folder with tiles

Index is created so that the user clicks using the right mouse button on the publish layer from which he wants to create tiles and chooses *WMTSTileEditor*. Here it is possible to create an index for a certain territory defined in the map window by drawing a polygon, which must then be selected.

In the dialog box, it is also possible to set since which minimal zoom will be tiles in the given area drawn. There is also displayed information with which layer is just being operated, if the tiles are cached and if the index file was created. Img 1 shows index drawn in the map window.

Info Icons

If the info icons are generated simultaneously with tiles, the map image rendering is not that much faster than rendering vectors without tiles, because it is necessary to load icons from the database. But if the info icons are hidden, the generating of map image is dependent only on the speed of loading tiles. Queries can then be called after activation of *PreSelect*.

Other Publish Tile Layer Setting Items

In category *6. OGC Web Map Tile Service* can be adjusted settings of the publish layer. Specifically the following items:

BackgroudColor – sets the tile background color

TileCleaner – category of other settings, it concerns cleaning (deleting) tiles. Invalidated (redrawn) are tiles incident with objects in form layers.

WmtsRootFolder – root folder for tiles of OGC WMTS service

WmtsTileExtent – sets the extension of limiting rectangle request for tiles in percent (for the text size in pixels)

Specific Settings for This Project

In this project is set tile generating for publish layer *_All*. This layer is activated by the popular layer button in the top left corner of the map window. After activation of this layer are tiles saved into the folder, which is defined in publish layer property in category *6. OGC Web Map Service*, item *WMTSRootFolder*. In the project is preset path: **c:/Marushkaexamples/Tile/test**, after displaying of the given layer in webserver is than in this path created folder with name corresponding to the GId of the given publish layer. In this folder are then created new subfolders with generated tiles.