# MGMaps Stored Maps format

Description for developers and map content providers



**Nutiteq LLC**
Register code: 11285684
VAT code: EE101099075
Bank account: Hansapank, 221032675017, code 767
Fix: (+372) 712 2334
Mob: +372) 509 2586
Address: Riia 181A, Tartu Science Park, Tartu 51014, Estonia

# Contents



# 1 About document

This is short description for MGMaps stored map data format. It could be used for creating new map content sets for customized MGMaps versions (which are based on MGMaps PRO SDK) or creating map content packages which can be added by end-user to the public general Mobile GMaps application.

## 1.1 Document history

| Who | When | Document revision | Comment |
|---|---|---|---|
| JaakL | 28.04.2008 | 1.1 | First version of the doc, describes map cache version 3. Based on description from Cristian Streng |

## 1.2 Compatibility

Format described in this document is applicable for MGMaps stored map cache **version 3**, which is compatible with MGMaps software versions, as minimum:

- Mobile GMaps versions: 1.38.0 to 1.39.15 (and possibly some newer versions)
- MGMaps Pro SDK kit release 1 (released 08.04.2008)

# 2 Map data format

Stored maps are using generally the same map tile image format (PNG files) and coordinate system numbering system (x, y and zoom) like on-line maps, see *MGMaps base map API* document about this. This document describes how the map tiles should be saved to the phone's internal memory or memory card to be readable by MGMaps client.

Note that there are following a bit different formats supported. Depending on data source most suitable should be chosen.

a) **Single tile per file** – each tile has own file.
   a. If map coverage area is big then number of files may grow too large which affects performance (performance is reduced significantly if flash drive has more than a few thousand files, it could be up to total instability of the phone)
   b. To reduce number of files per directory, use hash value as described below.

b) **Multiple tiles per file**– several tiles are stored to single file to reduce number of files and therefore improve performance

c) **MSR Mapcruncher format** – format used by a tool from research.microsoft.com/MapCruncher/ . This tool enables to georeference any map (e.g. scanned map) and then generate map tile files for different zooms. The map files can then be used as a overlay in a web map application which is using MSN Live! Maps API, and now also inside MGMaps client.

## 2.1 Directory structure

Directory structure for stored maps should be following:

*a) Single configuration file*

*MGMapsCache*/cache.conf

*b) Map content files, one directory per zoom level, 1..N files in directory:*

*MGMapsCache*/{map_type}_{zoom}/{mapfile}.mgm

Here:

- ***MGMapsCache*** - default map root directory name. Could be something else, spaces and special chars are not suggested in the name.
- **cache.conf** – stored map configuration file. It must have this name and format as described below
- **map_type** - Map Type identifying string, for own maps could be user selected string, but it should be consistent within the map set, and also it must match with the name defined in the Java application
- **zoom** – zoom level (0 to 16), see more about this in the on-line map API document
- **mapfile** – map file name, see below.

## 2.2 Cache configuration file

Basic stored map configuration values are written to **cache.conf** file. This file must have this name (Java client recognizes stored map directory by this file). The parameters in cache.conf are:

- **version** = 3 (current version of the cache format). Mandatory
- **tiles_per_file –** how many tiles per file, number. Mandatory.
- **hash_size** – see below. Default value 1
- **center** - initial position of the map when offline mode is enabled together with stored maps. This is used to automatically move the map to an area that was cached so that the user sees some map when the app starts (still if they have GPS enabled and have not downloaded maps for the correct area, the map will be moved back to the GPS position). Note that Map type here must be predefined map type, for example MicrosoftMap or YahooMap. Format:
    - center=<lat>,<longitude>,<zoom>,<map_type>
- **format** = mgmaps (default) or format = mapcruncher - whether the maps are stored in mgmaps or mapcruncher format
- **mapcrunchext** = png (default) or mapcrunchext = jpg - extension used by mapcruncher tiles (png or jpg format chosen in mapcruncher)

## 2.3 Map file naming and content

Map file naming and file contents depend on two parameters:

- tiles_per_file - how many tiles are stored to a file. It must be power of two. Suggested value is 16.
- hash_size – hash value used to create one more level of directories, to avoid too many files per single directory, which reduces performance and may create issues in some file systems. Use larger value if maps cover larger area.

Other parameters used below are:

- x, y – coordinates of tile, in Google's (projected) tile system. See on-line map API doc for details
- zoom – also from Google's coordinate system

File names:

| | Tiles_per_file = 1 | Tiles_per_file > 1 |
|---|---|---|
| Hash_size=1 | {mt}_{zoom0}/{x}_{y}.mgm | See below |
| Hash_size>1 | {mt}_{z}/**{h}/**{x}_{y}.mgm<br>h = (x*256+y)%hash_size | *Not supported currently* |

File content for single map per tile is just map tile in PNG or JPEG format, without extra headers.

## 2.4 Multi-tile file format

If there are multiple tiles per file, then file contents and naming is more complicated.

Let's name tiles-per-file value as **n**. Each file consists of a header of 6*n+2 bytes and the data. The first two bytes of the header specify the number of tiles currently stored, and the rest 6*n contain tile metadata. There are 6 bytes for each tile, 1 byte represents the dx coordinate, 1 byte the dy coordinate, and 4 bytes are the offset of the end of the tile data and beginning of next tile. All numbers are represented in big-endian format.

An example: n=32 , zoom=4, downloading tiles x=6, y=7 and x=7, y=7. Tile sizes are 12345 and 23456 bytes. We compute tiles-per-file-log = log2(n) = 5, then tiles-per-file-y-log = tiles-per-file-log/2 = 2 and tiles-per-file-x-log = tiles-per-file-log (minus) tiles-per-file-y-log = 3. Then tiles-per-file-x (tpfx) = 2^3 = 8, tiles-per-file-y (tpfy) = 2^2 = 4.

Now the file (where the tile is stored) name is built using trunc(x/tpfx) and trunc(y/tpfy), and dx=x%tpfx and dy=y%tpfy. In this example, both tiles are stored in the file 0_1.mgm with dx=6, dy=3 and dx=7, dy=3.

The offset where the first tile is known is already known, it's 6*n+2 (where the header ends). After dx and dy, the header contains the offset of the next file which is computed as offset of current tile + current tile size. For our example, 12345 = 3039h and 23456=5BA0h, 6*n+2 = C2h, C2h+3039h = 30FBh, 30FBh+5BA0h = 8C9Bh (the last offset is actually not used and it should represent the size of the entire file). The file data would then be:

```
00 02
06 03 00 00 30 FB
07 03 00 00 8C 9B
(tile 1 data)
(tile 2 data)
```

Tile tile data is original binary of the tile, in PNG or JPEG format.

### 2.5 Mapcuncher files

Mapcruncher map tiles are automatically named by the MSR Mapcruncher software, they use the same file names as the tiles on MSN virtual earth / Live Maps. Currently they are not re-organized in any way, they are just placed in a folder with a preferred map type name. See chapter below for more info about mapcruncher.

## 3 How to generate stored map files

There are some community-provided tools to generate stored maps from on-line map servers. The typical process involves two steps:

1. Create map area definition file.
2. Run map downloader script to read map content and store it to file

### 3.1 Step 1 – create map area definition file

Map area definition file defines map areas in different zoom levels, therefore it enables to define more detailed map coverage for some areas (like cities) and less detailed map for other areas. The file is simple test file with a .map extension and following contents:

```
<map type>=<map tile server URL>
<zoom min>-<zoom max>:<latitude south>, <longitude west> : <latitude north>, <longitude east>
<zoom min>… etc, repeat as many areas as needed
```

Example:

```
MyMap=http://lbs.nutiteq.ee/mt.php?
00-12: 51.77162806879235, 14.0625 : 55.144213289740016, 21.09375
00-12: 55.144213289740016, 15.46875 : 55.98735959497693, 18.28125
```

The Map Server URL must accept parameters x, y and zoom, which have same values like MGMaps on-line map server API (see separate document about this), and return one PNG file per tile.

Some notes:

- http://www.mapcacher.com/ is a community-created on-line tool to create map area definition file. User can define area as polygon, and the tool creates a map definition file (with .map extension). This tool cannot create different areas with different zooms and it creates map types for common web-based mapping servers. You should modify the map file manually to use your custom map server URL.
- Some useful .map files for some countries and cities are shared in community forum forum.mgmaps.com
- The .map file does not include any map data itself, and this file is not used by MGMaps Client directly. This file is only for defining area to be downloaded

## 3.2 Step 2 – download map data

For downloading actual map data MGMaps Pro SDK includes Perl script MapTileFE.pl, which requires also script called MapTileCacher.perl. Usage of this is following:

- Install and configure Perl. For Windows ActivePerl (http://www.activestate.com/Products/activeperl/) is suggested, it is available as free download.
- Configure MapTileCacher.perl file : review values for tiles_per_file (default 16) and hashSize values (default is 97, but if tiles_per_file>1, then it is taken as 1). Modify the script using text editor, the parameter values are in the beginning of the script.
- Run “MapTileFE.pl <.map definition file>”. If you do not specify .map file name then command-line script will ask one from the same directory where MapTileFE.pl is located
- Script will run and save the files. Depending on speed of the map server and network connection it may take few minutes up to several days.
    - If the script is stopped and later restarted with the same map definition file, then it will not re-download already downloaded files (if that these are still in the same subdirectory), and resumes from the cancelled point.

There also some community-provided GUI tools to help downloading, for example gMapMaker from Damien Debin (http://debin.net/gMapMaker/), see usage guide at http://forum.mgmaps.com/viewtopic.php?t=1116

# 4 Using MSR Mapcruncher

Starting with version 1.39.02, Mobile GMaps supports stored maps created using Microsoft's Map Cruncher tool. Follow these steps to make your own map and store it on your phone:

1. Download **MapCruncher** from http://research.microsoft.com/mapcruncher/ and install it.
2. Use MapCruncher to split your map into tiles. A nice tutorial is available at http://www.metacafe.com/watch/496160/5_minute_msr_mapcruncher_tutorial/
3. Open your output folder, then find the subfolder named **Layer_NewLayer** (or Layer_xxx where xxx is the name you gave the layer). You'll only be needing this subfolder, rename it as you wish (for example "MyMap").
4. Use following contents in the **cache.conf**. This is needed for MGMaps to recognize it as a stored map folder. When you do not modify the cache.conf, it will start centered somewhere in a corner of your downloaded map —you can change the "center" line to start centered on your actual map. The line includes 4 numbers meaning: latitude,longitude,zoom,maptype (you can leave maptype equal to MSRMap like given below or change it to some other map type which must be a preset map short-name or an already-added custom map).
5. Copy the subfolder and all the included files to your phone. You can remove all the other files and folders created by mapcruncher in the output folder.
6. On your phone, open MGMaps -> Settings -> Map Browsing and enable Stored Maps (and possibly Offline Mode too). Use the "Browse" option to browse to the folder with the new map. Save the settings
7. Define new custom map (MGMaps -> Settings -> Map Types, Add new custom map). It is important to use exactly the same map type name as *maptype* given in the cache.conf file, you can use the default *MSRMap* if you wish. You can left the custom map's URL empty.
8. Restart MGMaps to view your map.

**cache.conf** contents for MapCruncher maps:

```
version=3
format=mapcruncher
center=0.0,0.0,1,MSRMap
```

# 5 Copyright statements

Nutiteq supports only downloading map tiles from user's own custom map server.

It is user's obligation to follow copyright and usage terms of the map services and content they are using, check this before downloading and sharing any data. Most commercial public map servers have strict legal rules about downloading and using their content outside their own web browsing service, some others like OpenStreetmap and OpenAerialMap have more open policies.

Nutiteq does not provide or support any third party software or on-line tool mentioned above in this document. These are to be used at your own responsibility only.