MapTiler Manual Documentation

Release 0.6.3

Klokan Technologies GmbH

Jan 03, 2018

Contents

Contents:

CHAPTER 1

Welcome

MapTiler Pro is a software for map tile rendering. It transforms supplied raster geodata from existing coordinate system (SRS - Spatial Reference System) into map tiles suitable for Google Maps API mashups, native mobile applications with Apple MapKit, open-source RouteMe library for iOS or OSMDroid for Google Android platform. Supported is a direct export to Google Earth products as well. The produced tiles can be used for online publishing according the OpenGIS WMTS standard as well.

The produced map tiles can be easily served from existing in-house web servers, from practically any standard web-hosting provider and from a public or private cloud. Hosting of maps is also possible from an external content distribution network (such as the Akamai's CDN with over 100.000 servers in 78 countries) to serve the geodata with higher speed and reliability by automatically caching it geographically closer to your online visitors.

MapTiler can be used for processing large quantity of input files with high-resolution. The tool has been designed for producing seamless maps and aerial photo layers covering whole countries. The rendering is fast and efficient, and it can fully utilize multiple CPUs to 100%. It is heavily optimized and directly working with the raw input data and computer memory mapping for producing the map tiles - without any intermediate steps and layers (such as HTTP requests / transcoding of the rasters / etc).

This and other internal tricks makes this tool a magnitude faster then any other existing solution.

Thank you for choosing MapTiler.

CHAPTER 2

Installation

2.1 Windows

Download the appropriate setup.exe file and start the installation wizard. After it is finished a new section is added to Start -> All Programs and it is possible to use the command "maptiler" from the standard Command Prompt application in Windows.

2.2 Mac OS X

Download the installation disk image (DMG) and after opening the disk image in Mac OS X you should drag the "MapTiler Pro" icon into your "Application" folder. The command line interface can be started from Terminal application as: */Applications/MapTilerPro.app/Contents/MacOS/maptiler*.

All version of Mac OS X higher then 10.7 are supported.

2.3 Linux

Download the installation package (.deb or .rpm) for your Linux distribution. The packages are compiled against the standard out-of-the-box GDAL installation available for your Linux distribution (typically it is the package gdal automatically available via apt or yum).

In case a different version of the GDAL library is required for your project, we can provide you with binaries compiled for such a version on request.

Supported linux distributions are: Debian, Ubuntu, Redhat and Fedora (the rpms are dependent on GDAL from EPEL repository). Another option for running the latest MapTiler with the latest GDAL on any Linux distribution is via the Docker container.

CHAPTER 3

Usage

3.1 Getting started

The main purpose of the software is to generate tiles in a defined tiling system, given some options and inputs. The only mandatory option is -o for the output directory. This directory must not exist yet when you run MapTiler, to avoid overwriting existing data by mistake. As input, you can just provide the source dataset filename(s)[1].

```
maptiler -o output_directory input_file.ext
```

an example:

```
maptiler -o tiles map.tif
```

To render more files at once, just specify them one after another:

```
maptiler -o output_directory input1.tif input2.tif input3.tif
```

If you start the maptiler without arguments or with -help option, it will print all available commands:

```
maptiler -help
```

3.2 Output

The default behaviour of MapTiler is to write tiles each into its own file, under the directory structure:

output_directory/z/x/y.ext

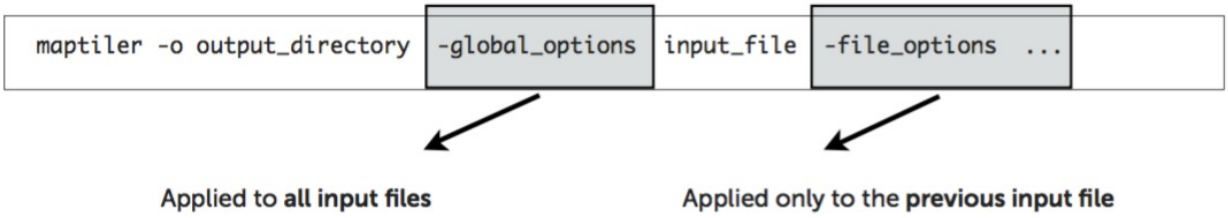
Where z is the zoom level, and x and y tile coordinates on relevant zoom level in the tiling profile.

The produced directory structure contains also a simple HTML viewer and description of the dataset in metadata.json, compatible with mb-util and TileServer project supporting OGC WMTS standard.

[1] Depending on your operating system you may need to call the command differently then just maptiler, typically on Linux and Mac in actual directory as ./maptiler and on Windows as maptiler.exe.

MapTiler supports direct output of the rendered map tiles into an SQLite database (MBTiles format). This simplifies transfer and management of the tilesets and is practical for mobile applications.

3.3 MapTiler Command Structure



The global options apply to all input files, in other words:

Only arguments specified BEFORE the input filenames are applied to all files!

Arguments which should be applied only to a single file are specified AFTER the name of such file (for example zoom level range specific only to that file) and has higher priority then the global options.

3.4 Available output options

3.4.1 Tiling profile / Tile Matrix Set

A global option defining the output system of tiles - the target coordinate system, tile pixel size, etc. MapTiler comes with three predefined most popular systems and possibility to specify a custom profile.

-mercator DEFAULT. The spherical mercator tile profile compatible with Google, Bing, Yahoo Maps, MapQuest, OpenStreetMap, and mobile maps on iOS and Android. This is the most commonly used profile. It uses coordinate system defined as EPSG:3857 or EPSG:900913. Details at: http://www.maptiler.org/google-maps-coordinates-tile-bounds-projection/[2].

In case you wish to use other tiling system, you must specify it as the first command on the command line. These are the alternatives:

-geodetic WGS84 Plate Caree / Unprojected. Compatible with most existing WMS servers, OpenLayers base map.

-gearth Tile profile specific for Google Earth according the KML SuperOverlay definition.

-custom You can specify your own tiling system. See the appropriate section under the Advanced options chapter for more information.

-raster Rendering raster files without a need of georeference is possible as well.

-garmin To produce format for Garmin GPS devices, with a size 1024x1024 pixels and output to .kml file. Afterwards, pack the tiles and the .kml tiles into a .zip archive, change its extension to .kmz and save it into the device.

Example: command for producing tiles for use with Google Earth:

```
maptiler -gearth -o tiles map.tif
```

-scale value To create high-resolution Retina / HiDPI tiles with variable scale. Retina tiles are available for each profile listed above.

[2] MapTiler uses Google XYZ naming of tiles, while older open-source MapTiler and GDAL2Tiles used the TMS naming (with flipped Y axis). In case you need the older TMS naming there is an option -tms for back compatibility.

Example: command for producing standard Retina tiles in mercator profile:

```
maptiler -mercator -scale 2.0 -o tiles@2x map.tif
```

Example: command for producing Retina tiles at 1.5 scale in raster profile:

```
maptiler -raster -scale 1.5 -o tiles-retina map.tif
```

3.4.2 Zoom levels

-zoom This option determines which layers of the tile pyramid will be generated. Default is the "native" level as calculated from image resolution. In case you need to add additional zoom levels, you can either define them as absolute numeric values or as relative numbers to the "native" levels with prefix + and -.

Each input file can have it's own explicit option for zoomlevels.

Example: zoom levels are automatically calculated as eg. 1 - 5

```
maptiler -o tiles map.tif
```

Example: zoom levels are explicitly set to be 3 - 5

```
maptiler -o tiles map.tif -zoom 3 5
```

Example: zoom levels are set to be 1 - 6 with relative value to native zoomlevels

```
maptiler -o tiles map.tif -zoom +0 +1
```

3.4.3 Tile formats

The produced tiles can be saved in one of several image format. MapTiler includes optimization of the final filesize and used number of colors (quantization), to minimize the disk size occupied by the rendered maps as well as the time necessary to transfer the maps to clients once the tiles are online.

Formats with support for transparency are:

-f png8a DEFAULT. Paletted RGBA PNG image.

-f png or -f png32 RGBA PNG image

-f webp or -f webp32 RGBA WebP image

Non-transparent formats are:

-f jpg or -f jpeg Progressive JPEG image in the YCbCr color space

-f png8 Paletted RGB PNG image

-f png24 RGB PNG image

-f webp24 RGB WebP image

3.4.4 Tile transparency or a background color

No matter what input datasets you specify, after transforming them into the tiling profile projection, MapTiler will handle them as RGBA images. The transparency can come from the image itself as an alpha channel (with support for

partly transparent areas), it can be derived from a selected color (so called NODATA color), or can be just a result of the transformation with the GDAL warping algorithm - for areas without available input data.

If the tile is completely transparent it is never saved to the disk to save the storage space.

If all of the pixels are fully visible (eg. opaque, maximum alpha is 255), the alpha channel is discarded and the tile is marked as non-transparent / opaque. Otherwise the tile is marked as partly transparent with alpha.

If partly transparent tiles are saved into a tile format without support for transparency (such as JPEG specified with -f jpg option) then the background color is applied. Default background color is white (255,255,255), but you can specify your own with the option:

-bg [r] [g] [b] The color of the background replacing transparency in the non-transparent tile formats.

For example:

```
maptiler -f png8 -bg 0 128 0 ...
```

-ignore_alpha If your dataset contains four channels, but the forth channel is not alpha channel, you can use this option for ignore this channel.

For example:

```
maptiler -f png32 -ignore_alpha input_4bands.tif ...
```

3.4.5 Tile store format

-store dir|mbtiles This option enforces the form of storage which is used for saving the rendered tiles. Possible options are the directory (dir) and the MBTiles (mbtiles). The default is the directory, but in case the -o parameter ends with .mbtiles then rendering into mbtiles is selected. This option specify the store form explicitely.

Note: for more details on this subject read the section Output in the chapter Usage above.

-sparse Skip the empty space between separate maps and don't create empty tiles. This option can improve speed of rendering, if there are huge areas between maps. This is default option for *-store dir*.

-no_sparse Fills the empty space between separate maps (if there is some) with empty tiles in background colour. This option can take longer to render and take more disk space, if there are huge areas between maps, as these have to be created. This is default option for *-store mbtiles*.

3.4.6 Hybrid tile format

MapTiler allows rendering into a hybrid tile format, so that transparent tiles are using transparent format (such as PNG) and tiles without any transparency at all are saved into a different format (such as JPEG). For aerial photos overlays or other datasets this can mean significant saving of the storage. Generated files are without extensions. This is done to simplify the generated OpenLayers viewer.

Example of usage:

```
maptiler -f hybrid <opaque> <transparent> ...
maptiler -f hybrid jpg png8a ...
```

3.4.7 Tile quality

There are some options to specify parameters of the conversion into image formats, which can significantly reduce size of produced tiles by degrading the output.

-jpg_quality The quality of JPEG compression. Number between 10 and 95. Default is 85.

-quant_quality The quality of quantization. Number between 1 and 100. Default is 100.

-quant_speed Higher speed levels disable expensive algorithms and reduce quantization precision. Speed 1 gives marginally better quality at significant CPU cost. Speed 10 has usually 5% lower quality, but is 8 times faster than speed 8. Default is 10.

If you experience issues with the visual quality of generated tiles with quantization involved try to set -quant_speed to lower values.

-webp_quality The quality of WebP compression. Number between 1 and 100. Level 100 means lossless compression. Default is 75.

-webp_alpha_quality The quality of WebP alpha channel compression. Number between 1 and 100. Level 100 means lossless compression. Default is 100.

Example of the rendering of a seamless map out of file map1.tif and map2.tif into tiles with internal palette with optimal colors with higher visual :

```
maptiler -o tiles -f png8a -quant_quality 90 -quant_speed 4 map1.tif map2.tif
```

3.4.8 Watermark

-watermark [image_file.png] It is possible to place your own watermark over rendered tiles to protect the online maps. The file should be smaller then a size of tiles. It is placed on a random position and burned into tiles.

A nice watermark file can be easily generated online by calling the Google Chart API: http://chart.apis.google.com/chart?chst=d_text_outline&chld=FFFFFF\T1\textbar{}11\T1\textbar{}h\T1\textbar{}000000\T1\textbar{}b\T1\textbar{}%C2%A9%20ABC

By replacing ABC in the end of this url a custom text phrase can be specified. We recommend to set the transparency of such watermark file by using a Photoshop or similar tool before applying it with MapTiler.

Example of usage of the watermark:

```
maptiler -o tiles -watermark watermark_image.png map.tif
```

3.5 The input files and related options

3.5.1 Supported input file formats

MapTiler is able to open and process large number of raster geodata formats, including: GeoTIFF, Erdas Imagine, ECW, MrSID, JPEG2000, SDTS, DTED, NITF, HDF4/5, BSB/KAP, OziExplorer, etc.

The complete list of supported formats is available online at: http://www.gdal.org/formats_list.html

3.5.2 Spatial reference system

Practically any modern existing georeferencing coordinate system (SRS - spatial reference system, e.g. geodetic datum + map projection with parameters) is supported, which means the software can process almost any geodata you may have available from all over the world.

In case the input files contains already the definition of used coordinate system (SRS) then MapTiler is able to load it and directly use this information for transformation of the maps. In case this information is missing in the supplied

file or it is incorrect (the maptiler place the maps on a wrong location, you can still assign the information about the spatial reference system with an option:

-srs [definition] Dataset projection. Can be WKT, EPSG code in the form of 'epsg:XXXX', PROJ.4 string. Beware of escaping. To search for identifiers or definitions use http://www.spatialreference.org/.

Example of assigning the United Kingdom spatial reference OSGB to a GeoTIFF file before rendering:

```
maptiler -o tiles -srs EPSG:27700 map_in_osgb.tif
```

3.5.3 Transparency from a color

-nodata [r] [g] [b] This command is typically used to eliminate borders of multiple map sheets that are stitched together. You can set a specific color of the map to be considered fully transparent during rendering.

Example for removing fully black border around a map:

```
maptiler -o tiles map.tif -nodata 0 0 0
```

3.5.4 Georeference / calibration

For proper rendering of the maps the location of supplied input files in the known coordinate system (SRS) must be available. MapTiler is loading the geolocation automatically from the internal headers of the input files (such as GeoTIFF) or from external supportive files (such as ESRI WorldFile) if they are avaialble.

To enforce a custom selected georeference information or loading from external files these options are available:

-bbox minx miny maxx maxy To manually set bounds of a file in the specified spatial reference system.

-geotransform posX scaleX rotX posY rotY scaleY To assign affine transformation directly. This option can be also used with its short name -gt.

-georeference [path_to_file] An option to load external georeference from World File, Tab File, OziExplorer Map File or .prj file.

-corners east1 north1 east2 north2 east3 north3 To assign affine transformation with 3 corner points: [0, 0], [width, 0], [width, height]. This option can be used with WGS84 Coordinate System (EPSG:4326) as arguments *lng1 lat1 lng2 lat2 lng3 lat3*, which will set up -srs EPSG:4326 for files without specified Coordinate system.

3.5.5 Cutline (Crop)

There are two command line options for cutline: -cutline and -cutline_proj. They specify the cutline (a clipping path) for an input image in pixels or in projected coordinates. They both expect a file name. The file can be either CSV or an OGR dataset (such as ESRI ShapeFile .shp).

From an OGR file, MapTiler will load all polygons and multi-polygons from all features of the first layer.

The CSV format with pixel coordinates of nodes of a triangle, more lines will create polygon:

X1,Y1

X2,Y2

X3,Y3

Example of use of such a pixel-based cutline:

```
maptiler -o outputdir input.tif -cutline polygon.csv
```

Another example of cutline with geocoordinates stored in a .shp file (may require accompanying .prj file with coordinate system):

```
maptiler -o outputdir input.tif -cutline_proj shape.shp
```

Embedded cutline can be ignored using option -cutline IGNORE

```
maptiler -o outputdir input_with_cutline.tif -cutline IGNORE
```

A cutline is specific for each input file - so the parameter should be used after a filename (see section MapTiler Command Structure).

3.5.6 Multiple files into multiple MBTiles or Folders

MapTiler is designed to produce a single merged layer from multiple input files. If you need to process multiple files and for each produce separate tileset then a batch processing is recommended.

Example:

This command processes every .tif file in a local directory and creates .mbtiles from each in the output directory. If .mbtiles is removed from the command, it produces separate directories instead. The command differs on operating systems:

Windows

```
for %f in (*.tif) do ( echo %f && maptiler -o output/%f.mbtiles %f )
```

When used in a batch file the %f must be %%f.

Linux / Mac OS X

```
for %f in *tif; do echo $f; maptiler -o output/`basename $f .tif`.mbtiles $f; done;
```

3.6 Advanced options

3.6.1 Options in the optfile

In case you have a large number of arguments to pass to maptiler, such as many input files (total amount is unlimited for maptiler), you can prepare a text file with all the arguments and call it with – optfile myarguments.txt. List of files can be easily created with ls or dir commands.

Any arguments normally passed on the command line could be part of the –optfile text file. Maptiler can combine arguments on the command line with arguments in the text file, such as:

```
maptiler -o output_directory --optfile myarguments.txt
```

3.6.2 Temporary directory location

During rendering, MapTiler also writes a substantial amount of data to a temporary directory. Not as much as will be in the output directory, but still. Please make sure there is enough space in the filesystem for it.

By default, the temporary directory will be created in the system default temporary location (*/tmp/* on Unix-like systems, or path from the environment variable%TEMP% on Windows-like systems). You can override this with the option:

-work_dir [directory] The location where to store temporary data during rendering. By default the system temporary directory.

Example:

```
maptiler -work_dir /tmp -o /mnt/data/tiles /mnt/maps/*.tif
```

3.6.3 Resampling methods

The visual quality of the output tiles is also defined by the resampling method. Selected method is used for interpolation of the values of individual pixels and it affects the sharpness vs smoothness of the produced maps.

-resampling near Nearest neighbor resampling. Rarely makes sense for production data. Can be useful for quick testing, since it is much faster the the others.

-resampling bilinear DEFAULT. Bilinear resampling (2x2 pixel kernel).

-resampling cubic Cubic convolution approximation (4x4 pixel kernel).

-resampling cubic_spline Cubic B-Spline Approximation (4x4 pixel kernel).

-resampling average Average resampling, computes the average of all non-NODATA contributing pixels. (GDAL >= 1.10.0)

-resampling mode Mode resampling, selects the value which appears most often of all the sampled points. (GDAL >= 1.10.0)

Resampling overviews produced by MapTiler are using average method, by default. Another possible method is Nearest neighbor.

-overviews_resampling near Nearest neighbor overviews resampling. Mostly used for elevation maps or similar.

-overviews_resampling average Average overviews resampling, computes the averate of all non-NODATA contributing pixels.

3.6.4 Defining a custom tiling profile for a specified coordinate system

MapTiler allows to define a custom system of tiles which should be rendered. Such tiling scheme, or in the terminology of OGC WMTS service the TileMatrixSet is for the maptiler defined with parameters which must follow the tile profile option: -custom.

-tiling_srs [definition] The spatial reference system, e.g. the coordinate system in which the tiles are created. Follows the definitions known from -srs.

-tiling_bbox [minx] [miny] [maxx] [maxy] The area which should be split into tiles defined in the tiling_srs coordinates.

-tiling_resolution [zoomlevel] [resolution] Resolution in units of the tiling spatial reference system per pixel on the given zoom level. MapTiler will automatically compute values for all other zoom levels, each having half the resolution of the previous one.

-tiling_resolution from_output Resolution is calculated so as to fit whole input mapset into one tile on zoom level 0 with respect to bbox, srs and tile size.

-tiling_resolution from_input Default behaviour if resolution is not specified. Resolution is calculated so as to not supersample the largest input map with respect to bbox, srs and tile size.

-tile_size [width] [height] The pixel dimmensions of one tile.

-tiling_centered Tile (0, 0) is in the center of the world.

3.6.5 Advanced warping arguments

The advanced warping algorithms parameters can be specified with the option:

-wo "NAME=VALUE" The warp options. See the papszWarpOptions field at http://gdal.org/ structGDALWarpOptions.html.

Example:

```
maptiler -o tiles -wo "SAMPLE_GRID=YES" t.tif -wo "SOURCE_EXTRA=16"
```

3.6.6 Usage on a computer cluster

MapTiler can run on an MPI cluster if a cluster specific binary has been requested. If you have the MPI version, a shell wrapper to run it on a cluster is delivered as well.

A version of MapTiler utilizing Map Reduce approach and Hadoop is under development, this will replace the older MPI.

More details are provided on request.

3.6.7 Merge MBTiles utility

The utility allows to update a previously rendered dataset and replace a small existing area with a different newly rendered raster data. Typical use-case is fixing of a small geographic area in a large seamed dataset previously rendered by MapTiler from many input files.

The utility also extent the bounding box of the tiles - it is usable for merging two just partly overlapping maps into one bigger map covering larger extent.

Usage:

```
merge_mbtiles [OPTION] BASE.mbtiles DETAIL.mbtiles [DETAIL_2.mbtiles]...
```

Typical usage:

1. render large dataset with MapTiler Pro - from several input files and produce large MBTiles (with JPEG or PNG tiles internally): *large.mbtiles*
2. if you want to update one of the previously rendered input files in the existing dataset render just this file into MBTiles - with the PNG32 format and zoom-levels on which you want it to appear in the large dataset. Save the new small MBTiles with just one file to *patch.mbtiles*

Example:

```
merge_mbtiles large.mbtiles patch.mbtiles
```

Existing tiles available in both *large.mbtiles* and the *patch.mbtiles* are going to be merged. On same zoomlevels, patch.mbtiles will replace the original large.mbtiles - so the *large.mbtiles* will be updated in-place.

Futher options:

-P n Set limit on defined number of cores.

-no_sparse Fills the empty space between separate maps (if there is some) with empty tiles in background colour. This option can take longer to render, if there are huge areas between maps, as these have to be created. In case the maps overlap each other, there is no extra action involved. Default behaviour without this option does not fill the empty space between separate maps.

-reencode This option is useful, when the 2 merged maps have different format (e.g. jpeg and png). By default, the result is a hybrid format (combination of both of them). If reencode option is used, the chosen file is encoded to the actual format (which can slow down the process).

CHAPTER 4

Hosting

4.1 Standard web server

On a standard hosting (such as an ordinary company web server) you can very simply host your maps. Just load the directory with tiles to your web hosting and the layer is automatically available.

Once uploaded, the produced maps can be opened in any viewer supporting OGC WMTS standard.

For hosting of MBTiles, you can use an open-souce TileServer (available at: https://github.com/klokantech/tileserver-php/), that can be used with any standard hosting that supports PHP. Upload the created maps and get dozens of popular web viewers with interactivity, including Google Maps API, Leaflet, OpenLayers, WebGL Earth and MapBox JS.

4.2 Cloud hosting

CloudPush command can be used for upload to Amazon S3 or Google Cloud Storage hosting. Examples are shown on the S3. If you need to use Google Cloud Storage, just change the "s3" to "google" or "gs".

Amazon access and secure key is available via IAM service administration interface.

The credentials for the Google Cloud Storage are under "Enable interoperable access" in the menu of the service.

Cloud Push instance is initialized with the first uploaded map via this command line utility. It automatically creates empty *index.json*, TileServer in *index.html* and sets WebSite configuration for this bucket.

Upload tiles from an MBTiles file to S3

```
cloudpush --acces_key ACCESS_KEY --secret_key SECRET_KEY s3://bucket_name add␣
↪filename.mbtiles
```

List all maps in the cloudpush tile storage

```
cloudpush --acces_key ACCESS_KEY --secret_key SECRET_KEY s3://bucket_name list
```

Delete a map

```
cloudpush --acces_key ACCESS_KEY --secret_key SECRET_KEY s3://bucket_name delete␣
↪filename
```

Delete whole cloudpush storage

```
cloudpush --acces_key ACCESS_KEY --secret_key SECRET_KEY s3://bucket_name destroy
```

Instead of providing the access credentials in every command these can be set as system environment variables:

```
set AWS_ACCESS_KEY_ID=[THE_ACCESS_KEY]
set AWS_SECRET_ACCESS_KEY=[THE_SECRET_KEY]
```

for Amazon S3, or

```
set GOOG_ACCESS_KEY_ID=[THE_ACCESS_KEY]
set GOOG_SECRET_ACCESS_KEY=[THE_SECRET_KEY]
```

for Google Cloud Storage,

and call the utility without these arguments:

```
cloudpush s3://bucket_name list
cloudpush s3://bucket_name add filename.mbtiles
```

It is possible to use further options such as:

--create-bucket	automatically creates bucket, if not existing
--no-index-json	not handling metadata in CloudPush instance index.json
--raw	same as –no-index-json
--basename <path>	sets custom basename (default: basename of MBTiles file)
--private	uploaded objects are private (default: public)

List of available parameters can be displayed by running ./cloudpush without any parameter

Example for using custom basename:

```
./cloudpush --basename myfile s3://bucket_name add filename.mbtiles
```

uploads tiles with URL format: *myfile/z/x/y.ext*. Custom basename contains directory separators (slash), for example:

```
./cloudpush --basename year/month/myfile s3://bucket_name add filename.mbtiles
```

result will have URL in format: *year/month/myfile/z/x/y.ext*.

Region-specific hosting can be set up via environment variable AWS_BUCKET_REGION=[value] or with parameter -R [value].

Example for EU (Ireland) region:

```
./cloudpush -R eu-west-1 s3://bucket_name add filename.mbtiles
```

List of S3 regions is provided by the utility with *–more-help* argument or visible at http://docs.aws.amazon.com/general/latest/gr/rande.html#s3_region

CHAPTER 5

Demo version

5.1 Setting CPU limits

MapTiler is a multi-threaded program. By default it will use all the CPUs you have and print the information about the cores. You can also set limit on defined number of cores yourself with the -P option.

```
maptiler -P 4 ...
```

This is especially practical to evaluate the demo application before purchasing a license for particular number of cores.

The modern CPUs has multiple cores and support of Hyperthreading - which provides multiple logical CPUs per core. This way MapTiler can provide higher performance with -P 4 even on a dual-core computer.

5.2 Demo trial extension and software activation online

The demo version of MapTiler is fully usable for 30 days after first start. In case this trial period for testing the software before purchase expires and you would like to continue to test the demo, it is necessary to contact Klokan Technologies and request a trial extension key.

Such key can be then used with the parameter:

-extend_trial KEY

And the demo can be then used during and extended time period.

After purchase of the software, when you receive your license key, it is necessary to activate MapTiler. After activation the demo version no longer enforce the "MapTiler DEMO" overlays.

To do the online activation, use the following command:

-activate YOUR-OWN-LICENSE-KEY

Example:

```
maptiler -activate YOUR-OWN-LICENSE-KEY
```

In case you require to reinstall the computer, use -deactivate **command to be able to re-activate the license later on.**

5.2.1 Software activation in a virtual machine

The activation process for MapTiler running in a virtual machine requires online activation only via environment variable *MAPTILER_LICENSE*. The software will be automatically deactivated in the end.

Example on Windows OS

```
set MAPTILER_LICENSE=YOUR-OWN-LICENSE-KEY
maptiler ...
```

Example on Linux / Mac OS X

```
export MAPTILER_LICENSE=YOUR-OWN-LICENSE-KEY
maptiler ...
```

Notice: This activation process via environment variable requires **not-activated MapTiler Pro** instance on that computer! Starting MapTiler application without setting this environment variable *MAPTILER_LICENSE*, you should see either **MapTiler Pro Demo**, or **Your trial period has expired** error message.

5.3 Software activation offline

For computers which are not directly connected to the Internet or which are in security restricted installations, we have support for offline activation as well.

To use the offline activation with the key we supply you after purchase you need to call:

```
maptiler -activate_request YOUR-OWN-LICENSE-KEY request.xml
```

This will generate a "request.xml" file, which you must send to us by email, and we will provide you with a "response.xml" file back. This can be used to activate the MapTiler by running:

```
maptiler -activate_response response.xml
```

If at any time you will want to deactivate MapTiler, eg. to move it to another machine, run the following command and you have to send us again the newly generated "request.xml" file.

```
maptiler -deactivate_request request.xml
```

CHAPTER 6

Indices and tables

- genindex
- modindex
- search